10-17-1994

Minimization of Generalized Reed-Muller Expansion and Its Sub-class

Xiaoqiang Zeng
Portland State University

THESIS APPROVAL

The abstract and thesis of Xiaoqiang Zeng for the Master of Science in Electrical and Computer Engineering were presented October 17, 1994, and accepted by the thesis committee and the department.

COMMITTEE APPROVALS:

[redacted]
Marek A. Perkowski, Chair

[redacted]
Michael A. Driscoll

[redacted]
Maria E. Balogh
Representative of the Office of Graduate Studies

DEPARTMENT APPROVAL:

[redacted]
Rolf Schaumann, Chair
Department of Electrical Engineering

* *

ACCEPTED FOR PORTLAND STATE UNIVERSITY BY THE LIBRARY

by [redacted] on 5 December 1994

ABSTRACT

An abstract of the thesis of Xiaoqiang Zeng for the Master of Science in Electrical and Computer Engineering presented October 17, 1994.

Title: Minimization of Generalized Reed-Muller Expansion and Its Sub-classes

Several classes of AND-EXOR circuit expressions have been defined and their relationship have been shown. A new class of AND-EXOR circuit, the Partially Mixed Polarity Reed-Muller Expression(PMPRM), which is a subclass of the Generalized Reed-Muller expression, is created, along with an efficient minimization algorithm. This new AND/EXOR circuit form has the following features:

- Since this sub-family of ESOP (with a total of $n2^{n-1}2^{2^{n-1}} - (n-1)2^n$ forms which includes the 2^n Fixed-Polarity Reed-Muller forms) is much larger than the Kronecker Reed-Muller(KRM) expansion(with 3^n forms), generally the minimal form of this expansion will be much closer to the minimal ESOP than the minimal form of KRM expansion.
- It is a sub-class of the Generalized Reed-Muller Expansion, thus has better testibility than other AND/EXOR circuits. Those design methods of easily testable GRM circuit networks[6][35] can also be used for this new circuit form.
- The exact solution to the minimization of this new expansion provides a upper-bound for the minimization of GRM expansion.

In this thesis, we prove that to calculate a PMPRM expansion from one of its

adjacent polarity expansion , only one EXOR operation is needed. By calculating the adjacent polarity expansions one-by-one and searching all the PMPRM forms the minimum one can be found. A speedup approach allows us to find the exact minimum PMPRM without calculating all forms. The algorithm is explained by minimizing the 3-variable functions and is demonstrated by flow graphs.

With the introduction of termwise complementary expansion diagram, a computerized algorithm for the calculation of any GRM expansion is presented. The exact minimum GRM form can be obtained by an exhaustive search through all GRM forms. A heuristic minimization algorithm, which is designed to decrease the time complexity of the exact one, is also presented in this thesis. Instead of depending on the number of input variables, the computation time of this quasi-minimum algorithm depends mainly on the complexity of the input functions, thus can solve much larger problems.

The exact minimization algorithm for PMPRM and the quasi-minimum GRM minimization algorithm have been implemented in C programs and a set of benchmark functions has been tested. The results are compared to those from [16], [36], and Espresso's. In most cases our program gives the same or better solutions.

MINIMIZATION OF GENERALIZED REED-MULLER EXPANSION AND ITS SUB-CLASSES

by

XIAOQIANG ZENG

A thesis submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE
in
ELECTRICAL AND COMPUTER ENGINEERING

Portland State University
1994

ACKNOWLEDGEMENTS

A number of people have assisted me in the research reported in this thesis. I would like to take this opportunity to thank them.

I would like to thank Marek A. Perkowski, Chair of the Committee, for his initial inspiration and professional guidance throughout the research.

I would also like to thank other Committee members: Michael A. Driscoll and Maria E. Balogh for their assistance and helpful suggestions in the preparation of the thesis.

I am grateful to Shirley Clark and Laura Riddell for their provision of all kinds of help during my research period.

My friend, Haomin Wu, gave me so much help throughout my study and research in Portland State University. I owe him a lot.

Finally, I am deelply appreciative of the patience, encouragement and companionship of my wife, Si. Without her love and inspiration, motivation would be lacking to complete this project.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF TABLES

LIST OF FIGURES

CHAPTER I

INTRODUCTION

With the development of VLSI technology, the Exclusive-OR Sum of Products(ESOPs) circuits become increasingly attractive due to its following advantages:

(1) ESOPs circuits are generally more economical than Sum of Product circuits[1-4]. Sasao's approach in [5] confirms that programmable logic arrays(PLAs) implementing ESOPs of randomly generated functions required, on the average, fewer product terms than the standard PLAs implementing SOPs.

(2) The ESOPs circuits, especially their canonical sub-classes, have high testability[6][7], and are especially suitable for testable VLSI design.

(3) Many practical circuits, especially in arithmetic and telecommunication networks, are heavily EXOR oriented.

(4) The synthesis techniques of ESOPs can be readily extended to multiple-valued switching circuits owing to the modular structure of ESOPs.

The slow speed and large chip area of EXOR gate were the major obstacles to the wide use of ESOPs in the past. Owing to these disadvantages, the PLAs implementing ESOPs were not put into practical use for a long time despite of their many advantages. With the arrival of FPGAs devices, this deficiency no longer holds, and the theories developed for instance in [6][7] can be practically used. Several families of new PLD devices that have been recently marketed: Table Look-up based(Xilinx) and Multi-

plexer based(Actel 1020) Field Programmable Gate Arrays, folded NAND devices (Signetics LHS501), and EXOR PLDs, either directly include EXOR gates(LHS501) or allow them to be realized in "universal modules". Since the five-input EXOR gate in Xilinx has the same speed and area cost as, for instance, a five-input OR gate, new design methods are needed for such technologies that assume the usage of EXOR gates on the same full rights as the AND and OR gates.

Of particular interest are the AT6000 series FPGAs from ATMEL Corporation[12], these devices are truly cellular FPGAs. Each logic function may be constructed from various two-input gates such as EXORs, ANDs, NANDs and Inverters. The ESOP based logic is a prime candidate for these devices since an ESOPs realized Boolean function can be well mapped into these devices without any routing problem[9].

With the advent of these cellular FPGAs, the industry's demand for efficient ESOP minimizers greatly increases. Lack of an effective ESOP minimizer becomes now a major obstacle to the wider use of ESOP circuits.

The problem of finding the minimal ESOP of a Boolean function is classically a hard one in logic synthesis theory. So far, exact minimal solutions for ESOP can be practically found only for functions with not more than 5 variables because tremendous computation has to be involved[13]. Papakonstantinou[24] gave an exact algorithm for 4 input functions. The algorithm from [13] has theoretically no limits on the number of variables but 4 is its practical limit. Since exact solutions can be practically found only for functions with not more than 5 variables, the interest is in approximate solutions. Two approaches to generate sub-optimal solutions can be found in the literature. One approach is to minimize sub-families of ESOPs. Another approach is to minimize ESOPs using heuristic algorithms. Efficient programs for sub-families of ESOPs were given in [22][25]. Heuristic computer programs have been presented in

[3][4][7][27][28]. In these programs, two general methods are used. One method is to minimize the coefficients of Reed-Muller forms [3][25][26]. Another method is to perform a set of cube operations iteratively on minterms or disjoint cubes. An interesting approach is to minimize canonical sub-families of ESOP[14][15].

As sub-families of ESOP, Reed-Muller canonical forms are of special interests because they have high testability, and are especially suitable for testable VLSI design[32]. The existence of a very large number of Reed-Muller canonical forms is demonstrated in [14][15]. Among which exact and efficient algorithms have been created, however, only for the Fixed-Polarity Reed-Muller(FPRM) expansion (with the number of forms 2^n), the Kronecker Reed-Muller expansion(KRM)(with the number of forms 3^n) [1][16][18][20], and a few others[33].

In principle the minimal FPRM form is more complex than the minimal ESOP. Since the KRM forms are more general than the FPRM forms, the minimal KRM solution will be better than the minimal FPRM one. To our knowledge, no synthesis method was proposed for the exact minimization for other larger families of canonical forms. The main reasons are that they would involve a prodigious computation if no high-efficiency computation methods were found.

There is another Reed-Muller canonical expansion called Inconsistent Mixed-Polarity Reed-Muller(IMPRM)forms($2^n 2^{n-1} - 2^n$) identified by Cohn[21], which are produced by taking the zero-polarity RM form and inverting any combination of literals. Cohn's IMPRM forms exclude the consistent FPRM forms(2^n). Combining the IMPRM forms and FPRM forms will create the Generalized Reed-Muller (GRM) forms[15][21][22]. Mathematically, the GRM forms do not exhibit a general structure in the nested hierarchy of families of canonical forms in [14] because they are not constructed from Kronecker matrix products[20]. Therefore, no other computation methods have been proposed for them, except for the heuristic ones proposed by our group[22].

However, from the engineering point of view, the GRM forms($2^n 2^{n-1}$) are interesting because they contain many more forms than KRM forms(3^n) when $n \geq 2$. Hence, generally, the minimal GRM form will be closer to the minimal form of ESOP than the minimal KRM form. However, an exhaustive search for a minimal GRM form is also computationally unfeasible owing to the large number of forms.

In this thesis, a new canonical AND/XOR circuit form, defined as a Partially Mixed Polarity Reed-Muller(PMPRM) expansion, which is a canonical sub-family of Exclusive Sum of Products(ESOP), is created, along with its fast computation algorithm. This new AND/EXOR circuit form has the following features:

- Since this sub-family of ESOP (with a total of $n2^{n-1}2^{2^{n-1}} - (n-1)2^n$ forms which include the 2^n Fixed-Polarity Reed-Muller forms) is much larger than the Kronecker Reed-Muller(KRM) expansion(with 3^n forms), generally the minimal form of this expansion will be much closer to the minimal ESOP than the minimal form of KRM expansion.
- It is a sub-class of the Generalized Reed-Muller Expansion, thus has better testability than other AND/EXOR circuits. Those design methods of easily testable GRM circuit networks[6][35] can also be used for this new circuit form.
- The exact solution to the minimization of this new expansion provides a upper-bound for the minimization of GRM expansion.

By presenting the adjacent polarity GRM expansion calculation algorithm, the minimization of GRM form is also discussed. The experimental results are very encouraging.

Chapter II presents seven classes of AND/EXOR expressions: positive polarity Reed-Muller expressions, fixed polarity Reed-Muller expressions, Kronecker expressions, pseudo Reed-Muller expressions, pseudo Kronecker expressions, generalized Re-

ed-Muller expressions, and exclusive-or sum-of-products expressions(ESOPs). Relations among these classes are shown. Since this thesis is devoted to algorithms for the minimization of GRM expansion and its sub-classes, in Chapter III, we discuss the properties of this expansion in more details.

The new AND/XOR circuit form proposed in this thesis, the Partially Mixed Polarity Reed-Muller(PMPRM) expansion, is presented in Chapter IV. The Adjacent Polarity PMPRM Expansions are defined. In this chapter, we prove that to calculate a PMPRM expansion from one of its adjacent polarity expansion, only one EXOR operation is needed. By calculating the adjacent polarity expansions one-by-one and searching all the PMPRM forms the minimum one can be found. A speedup approach allows us to find the exact minimum PMPRM without calculating all forms. The algorithm is explained by minimizing the 3-variable functions and is demonstrated by flow graphs.

In Chapter V, the minimization of Generalized Reed-Muller expressions is discussed. First, we present some basic concepts and definitions. Then the termwise complementary expansion diagram is introduced. Using this diagram, one can calculate any GRM expansion with expected polarity from another given GRM expansion. The exact minimum GRM form can be obtained by an exhaustive search through all GRM forms. A heuristic minimization algorithm, which is designed to decrease the time complexity of the exact one, is also presented in this chapter. Instead of depending on the number of input variables, the computation time of this quasi-minimum algorithm depends mainly on the complexity of the input functions, thus can solve much larger problems.

In Chapter VI, the program is given and the results are shown. Chapter VII gives the conclusion.

CHAPTER II

VARIOUS CLASSES OF AND/EXOR CIRCUIT EXPRESSIONS

Many researchers defined various classes of AND-EXOR expressions[15][34], but the terminology is not unified. In this chapter, we give some basic concept and theory, define several classes of AND-EXOR expressions and show the relations among them. Since we are going to give the algorithm for the minimization of Generalized Reed-Muller expansion and its sub-classes, in the next chapter we will discuss the properties of this class in more details.

II.1 Basic Concepts

Definition 3.1: Literal $\dot{x}_i$ is a variable x_i in either positive($\dot{x}_i = x_i$) or negative ($\dot{x}_i = \bar{x}_i$) form.

Definition 3.2: Literal x_i is the complementary literal of literal $\bar{x}_i$. Literal $\bar{x}_i$ is the complementary literal of literal x_i.

Definition 3.3: A product of distinct literals is called a product term (product or term in short).

Definition 3.4: T_1 and T_2 are two terms. If T_1 and T_2 contain exactly the same literals, we say T_1 and T_2 are identical to each other.

Theorem 2.1 will describe the properties of operator $\oplus$. It will follow without proofs of these properties since they are well known[15][31].

Theorem 2.1: The following identities are true:

$$x_1 \oplus x_2 = x_1\bar{x}_2 + \bar{x}_1 x_2$$

$$x_1 \oplus x_1 = 0$$

$$x_1 \oplus \bar{x}_1 = 1$$

$$x_1 \oplus 0 = x_1$$

$$x_1 \oplus 1 = \bar{x}_1$$

$$x_1 \oplus (x_2 \oplus x_3) = (x_1 \oplus x_2) \oplus x_3$$

$$x_1(x_2 \oplus x_3) = x_1x_2 \oplus x_1x_3$$

Theorem 2.2(Expansion Theorem) An arbitrary logic function $f(x_1, x_2, \cdots, x_n)$ can be represented in any of the following forms:

$$f = 1 \cdot f_0 \oplus x_1 \cdot f_2 \tag{2.1}$$

$$f = \bar{x}_1 \cdot f_2 \oplus 1 \cdot f_1 \tag{2.2}$$

$$f = \bar{x}_1 \cdot f_0 \oplus x_1 \cdot f_1 \tag{2.3}$$

where $f_0 = f(0, x_2, x_3, \cdots, x_n)$, $f_1 = f(1, x_2, x_3, \cdots, x_n)$, and $f_2 = f_0 \oplus f_1$.

II.2 AND/EXOR Circuit Expressions

In the case of SOPs we can use only the type (2.3) expansion, which is often called a Shannon expansion. However, in the case of AND-EXOR expressions, we may use any of the three expansions. Thus, various classes of expressions exist as described in the following sections.

2.2.1 Positive Polarity Reed-Muller Expansion(PPRM)

When we apply the type (2.1) expansion to all the variables, we obtain an expression consisting of positive literals only:

$$f = a_0 \oplus a_1 x_1 \oplus \cdots \oplus a_n x_n \oplus a_{12} x_1 x_2 \oplus a_{13} x_1 x_3 \oplus \cdots \oplus a_{nn-1} x_n x_{n-1} \oplus \cdots \oplus \oplus a_{12\ldots n} x_1 x_2 \cdots x_n \quad (2.4)$$

This is defined as a Canonical Reed-Muller expansion. Since all the literals in this expansion are positive, it is also called Positive Polarity Reed-Muller Expression(PPRM). Because PPRM is unique for a given function, no minimization problem exists. The average number of product terms in the PPRMs for the n-variable functions is 2^{n-1}[5].

2.2.2 Fixed Polarity Reed-Muller Expression(FPRM)

When we apply either the type(2.1) or the type(2.2) expansion to each variable, we obtain an expression similar to (2.4), except that either a true or a complemented literal is used for each variable. This expression is called a Fixed Polarity Reed-Muller expression(FPRM).

There are at most 2^n different FPRMs for an n-variable function. The minimization problem is to find an expression with the minimum number of products among the 2^n different FPRMs. As for minimization, two different methods are known: one requires the space and the computation time of $O(2^n)$ and $O(4^n)$, respectively[1], and the other requires the space and the computation time of $O(3^n)$[15]. Many programs have been developed for this FPRM circuit form[15][18].

2.2.3 Generalized Reed-Muller Expression(GRM)

In the expression of the type (2.4), if we can freely choose the polarities of the literals, then we have a more general expression than a FPRM. This is called a Generalized Reed-Muller Expression(GRM)[15]. It is also called an inconsistent canonical form[18] or a canonical restricted mixed polarity form[22]. There are at most $2^n 2^{n-1}$ different GRMs. A heuristic minimization method is shown in [22]. Note that some researchers use the term GRMs to refer a different class of AND/EXOR expressions.

2.2.4 Kronecker Expression(KRO)

When we apply either the type(2.1) or (2.2) or (2.3) expansion to each variable, we obtain an expression which is more general than FPRM. This is called a Kronecker expression(KRO) since it can be represented by the Kronecker product[15]. There are at most 3^n different KROs for an n-variable function. As an algorithm to find a KRO with the minimum number of products, a method using an extended truth table of 3^n entries and the extended weight vector is known. The time and space complexity of the algorithm are $O(n3^n)$ and $O(3^n)$, respectively[15].

2.2.5 Pseudo Reed-Muller Expression(PSDRM)

When we apply either the type (2.1) or the type (2.2) expansion to f, we have two sub-functions. For each sub-function, we can apply either type (2.1) or type (2.2) expansion. However, assume that we can use different expansions for each sub-function. In this case, we have a more general expansion than a FPRM. This is called a Pseudo Reed-Muller Expression(PSDRM). In PSDRM, both true and complemented literals can appear for the same variable. There are at most 2^{2^n-1} different PSDRMs. A minimum PSDRM can be obtained from the extended truth table. However the number of products in the expression depends on the order of the variables. This class of expressions has not been studied according to the author's knowledge.

2.2.6 Pseudo Kronecker Expression(PSDKRO)

When we apply either the type (2.1) or (2.2) or (2.3) expansion to f, we have two sub-functions. For each sub-function, we can apply either the type(2.1), (2.2) or (2.3) expansion, and assume that we can use different expansions for each sub-function. In this case, we have a more general expansion than a KRO. This is called a Pseudo Kronecker Expression(PSDKRO)[15]. In PSDKRO, both true and complemented literals can appear for the same variable. There are at most $3^{2^n}-1$ different PSDKROs. A minimum PSDKRO can be obtained from an extended truth table. The number of products in the expression depends on the order of the variables.

2.2.7 Exclusive-or Sum-of-Products Expression(ESOP)

Arbitrary product terms combined by EXORs are called an Exclusive-or Sum-of-Products Expression(ESOP). The ESOP is the most general AND-EXOR ex-

pression. There are at most 2^{3^n} different ESOPs, where n is the number of the input variables. No efficient minimization method is known, and iterative improvement methods are used to obtain near minimal solutions[4][5][27][29][30]. An exact minimization method has been developed, but it is very time and memory-consuming[13].

II.3 Relations among the Classes

Theorem 2.2. Suppose that PPRM, FPRM, PSDRM, KRO, PSDKRO, GRM, and ESOP denote the set of expressions. Then the following relations hold:

①	PPRM	$\subset$	FPRM	②	FPRM	$\subset$	PSDRM
③	FPRM	$\subset$	KRO	④	KRO	$\subset$	PSDKRO
⑤	PSDRM	$\subset$	PSDKRO	⑥	PSDRM	$\subset$	GRM

(Proof) As for 1 --- 5, they are trivial and they follow from the definitions. As for 6, consider a PSDRM, It is also a GRM, and hence the proof is completed.

Example 2.1

$xy \oplus yz \oplus xz$ is a PPRM (all the literals are positive).

$x\bar{y} \oplus \bar{y}z \oplus zx$ is a FPRM, but not a PPRM.

(x and z have positive literals, but y has negative literals.)

$xy \oplus \bar{y}z \oplus \bar{z}x$ is a PSDRM, but not a FPRM.

(y and z have literals of both polarities).

$xyz \oplus \bar{x}\,\bar{y}\,\bar{z}$ is a KRO, but not a FPRM.

(x, y, z have literals of both polarities).

$\bar{x} \oplus xy \oplus x\bar{y}$ is a PSDKRO, but not a KRO.

$\bar{x} \oplus xy \oplus x\bar{y}$ is a PSDKRO, but not a PSDRM.

(it contains two products of the highest degree).

$x \oplus y \oplus \bar{x}\,\bar{y}$ is a GRM, but not a PSDRM.

$x \oplus y \oplus \bar{x}\,\bar{y}$ is a GRM, but not a PSDKRO.

$xyz \oplus \bar{x}\,\bar{y}\,\bar{z}$ is a KRO, but not a GRM.

(it contains two products of the highest degree).

$x \oplus y \oplus xy \oplus \bar{x}\,\bar{y}$ is an ESOP, but neither GRM nor PSDKRO.

The relations among the classes of expressions are shown in Figure 1.

From the above, we see that the number of GRM forms for a given logic function is much larger than that of the other classes of AND/EXOR expressions except the ESOP one. Therefore, the cost of the minimum ESOP form is usually much closer to the cost of the minimum GRM form than to the cost of the minimum form of other classes of expressions of these functions. This reason causes the superiority of the method proposed in this thesis with respect to the methods of the other minimization algorithms. Moreover, observe that the minimal FPRM form is always not better than the minimal GRM form.

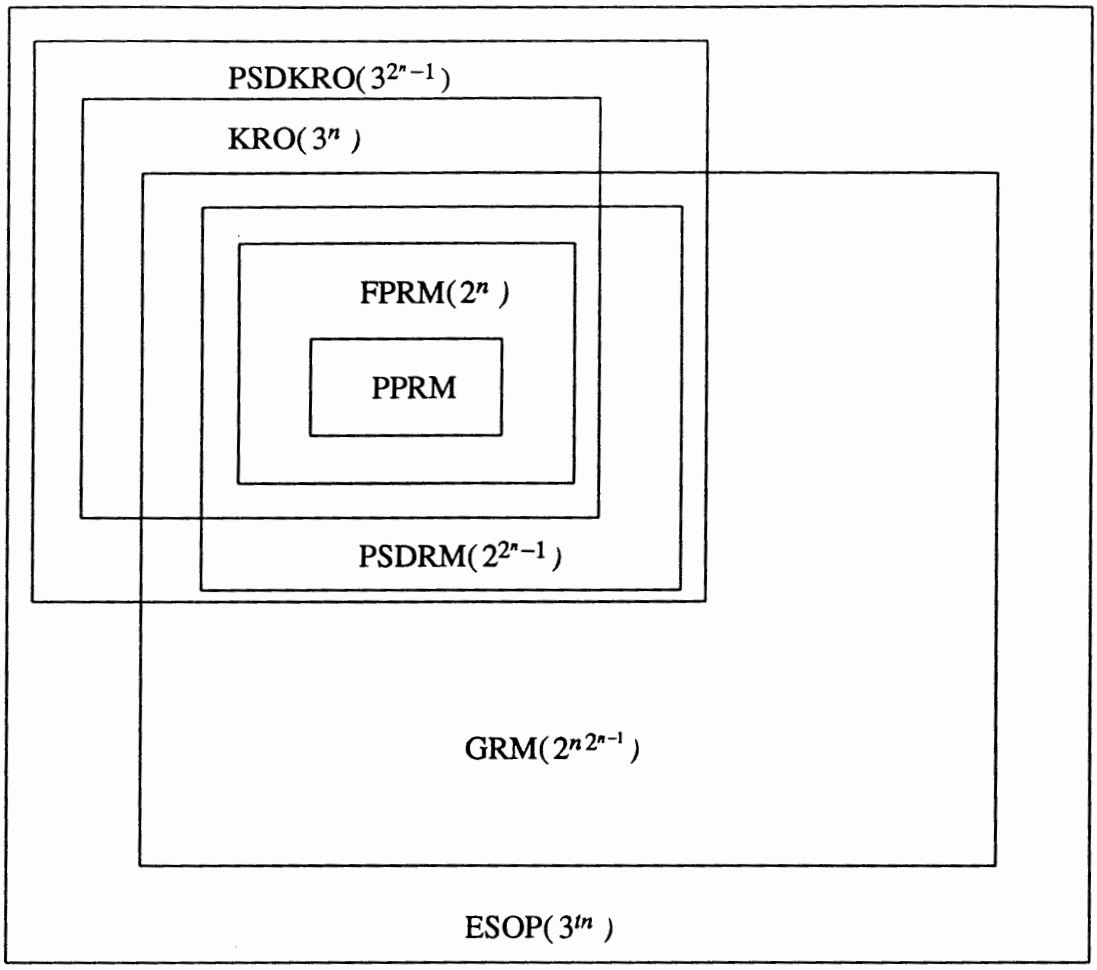


Figure 1. Relations among various classes of AND/EXOR Expressions

CHAPTER III

GENERALIZED REED-MULLER EXPRESSION

Mathematically, the Generalized Reed-Muller expansions do not exhibit a general structure in the nested hierarchy of families of canonical forms in [14] because they are not constructed from Kronecker matrix products[20]. However, from the engineering point of view, the GRM forms($2^{n2^{n-1}}$) are interesting because they contain many more forms than the KRM forms (3^n) when $n \geq 2$. Hence, generally the minimum GRM form will be closer to the minimal form of ESOP than the minimal KRM form.

III.1 Canonical Reed-Muller Expansion

When we apply type(2.1) expansion to all the variables of a logic function f, we have an expression consisting of positive literals only. For example, consider a 3-variable function $f(x_1, x_2, x_3)$. As shown in Theorem 2.2, the function $f(x_1, x_2, x_3)$ can be expanded as the following:

$$f(x_1, x_2, x_3) = f_0(x_2, x_3) \oplus x_1 f_2(x_2, x_3)$$

where $f_0(x_2, x_3) = f(0, x_2, x_3)$, $f_1(x_2, x_3) = f(1, x_2, x_3)$, and $f_2 = f_0 \oplus f_1$. Similarly, f_0 and f_2 are expanded as follows:

$$f_0(x_2, x_3) = f_{00}(x_3) \oplus x_2 f_{02}(x_3)$$

$$f_2((x_2,x_3)=f_{20}(x_3)\oplus x_2 f_{22}(x_3)$$

$$f_{00}(x_3)=f_{000}\oplus x_3 f_{002}$$

$$f_{02}(x_3)=f_{020}\oplus x_3 f_{022}$$

finally, we have

$$f=f_{000}\oplus x_3 f_{002}\oplus x_2 f_{020}\oplus x_2 x_3 f_{022}\oplus x_1 f_{200}\oplus x_1 x_3 f_{202}\oplus x_1 x_2 f_{220}\oplus x_1 x_2 x_3 f_{222}$$
$$=b_0\oplus b_1 x_3\oplus b_2 x_2\oplus b_3 x_2 x_3\oplus b_4 x_1\oplus b_5 x_1 x_3\oplus b_6 x_1 x_2\oplus b_7 x_1 x_2 x_3$$

The expansion of a n-variable function can be written as follows:

$$f(x_1,x_2,\cdots,x_i,\cdots,x_n)=b_0 x_1^0 x_2^0\cdots x_i^0\cdots x_n^0\oplus b_1 x_1^0 x_2^0\cdots x_i^0\cdots x_n^1\oplus\cdots\oplus$$
$$b_j x_1^{e_1} x_2^{e_2}\cdots x_i^{e_i}\cdots x_n^{e_n}\oplus\cdots\oplus b_{2^n-1} x_1^1 x_2^1\cdots x_i^1\cdots x_n^1 \quad (3.1)$$

where $x_i^0=1$, $x_i^1=x_i$,(i=1,2,$\cdots$,n); $b_j\in\{0,1\}$, (j=0,1,2,$\cdots$, 2^n-1), $e_i\in\{0,1\}$, and $(j)_{10}=(e_1 e_2\cdots e_i\cdots e_n)_2$.

This is defined as the Canonical Reed-Muller Expansion. Since all literals in this expression are positive, Eqn.(3.1) is also called a Positive Polarity Reed-Muller(PPRM) expression.

III.2 Generalized Reed-Muller Expression

Definition 3.1(a): the GRM forms are created by selecting any combination of literals in the Positive Polarity(or zero polarity) RM expansion and replacing these literals with their inverses[21][22].

From Eqn.(3.1) we observe that the PPRM expansion has 2^n terms. Each term is

a product of a subset of the n variables. This subset can be identified by the ones in the corresponding n-bit binary numbers $(e_1 e_2 \cdots e_n)_2 = (j)_{10}$. Thus, the following definition complies with definition 3.1(a).

Definition 3.1(b): The form of Eqn.(3.1) in which each variable can be both positive and negative but in which there is exactly one coefficient for each subset of variables of a term will be called a *Generalized Reed-Muller* expansion.

It follows from the preceeding definitions that the FPRM class is properly included in the GRM class.

Definition 3.2: By a Fixed-Polarity Reed-Muller expansion(FPRM) one denotes a form of GRM in which the literals of a variable are either positive or negative, and cannot stand in both forms in the same expression.

Definition 3.3: A Reed-Muller expansion (also called Positive Polarity Reed-Muller form) is a GRM form that consists of only positive literals.

Eqn.(3.1) can be used as the n-variable Generalized Reed-Muller expansion except for every literal $\dot{x}_i$ is used instead of x_i where for each $\dot{x}_i$ in different terms we have $\dot{x}_i = x_i$ or $\dot{x}_i = \overline{x_i}$. If we use T_j to represent product term $b_j \dot{x}_1^{e_1} \dot{x}_2^{e_2} \cdots \dot{x}_i^{e_i} \cdots \dot{x}_n^{e_n}$, then the GRM expression can be written as

$$f = T_0 \oplus T_1 \oplus \cdots \oplus T_j \oplus \cdots \oplus T_{2^n-1}$$

Please note, that some researchers use

$$f = a_0 \oplus a_1 \dot{x}_1 \oplus \cdots \oplus a_n \dot{x}_n \oplus a_{12} \dot{x}_1 \dot{x}_2 \oplus a_{13} \dot{x}_1 \dot{x}_3 \oplus \cdots \oplus a_{n-1} \dot{x}_n \dot{x}_{n-1} \oplus \cdots \oplus$$
$$\oplus a_{12\cdots n} \dot{x}_1 \dot{x}_2 \cdots \dot{x}_n$$

to represent a GRM form. This is the same as Eqn. 3.1 except that the terms are in different positions. In this thesis we always use Eqn. 3.1 as the GRM expansion.

Definition 3.4: In a GRM expansion, any literal $\dot{x}_i$ can be expressed as $x_i \oplus \delta$ where $\delta \in \{1,0\}$. Here δ is defined as the polarity of literal $\dot{x}_i$.

$$\delta = \begin{cases} 0 & \dot{x}_i = x_i \quad (positive) \\ 1 & \dot{x}_i = \bar{x}_i \quad (negative) \\ - & \dot{x}_i = x_i \text{ or } \dot{x}_i = \bar{x}_i \quad (positive \text{ or } negative) \end{cases}$$

Example 3.1: A GRM form of a 3-variable function is as follows:

$$f = f_e = x_2 \oplus \bar{x}_2 x_3 \oplus \bar{x}_1 \bar{x}_2 x_3$$

This expression can be written as:

$$f_e = 0 \oplus 0\cdot(x_3 \oplus -) \oplus 1\cdot(x_2 \oplus 0) \oplus 1\cdot(x_2 \oplus 1)(x_3 \oplus 0) \oplus 0\cdot(x_1 \oplus -) \oplus 0\cdot(x_1 \oplus -)(x_3 \oplus -)$$
$$\oplus 0\cdot(x_1 \oplus -)(x_2 \oplus -) \oplus 1\cdot(x_1 \oplus 1)(x_2 \oplus 1)(x_3 \oplus 0)$$

$$= T_0 \oplus T_1 \oplus T_2 \oplus T_3 \oplus T_4 \oplus T_5 \oplus T_6 \oplus T_7$$

Where

$T_0 = 0$

$T_1 = 0\cdot(x_3 \oplus -)$

$T_2 = 1\cdot(x_2 \oplus 0)$

$T_3 = 1\cdot(x_2 \oplus 1)(x_3 \oplus 0)$

$T_4 = 0\cdot(x_1 \oplus -)$

$T_5 = 0\cdot(x_1 \oplus -)(x_3 \oplus -)$

$T_6 = 0\cdot(x_1 + -)(x_2 \oplus -)$

$T_7 = 1\cdot(x_1 \oplus 1)(x_2 \oplus 1)(x_3 \oplus 0)$

In the above GRM expansion, if the coefficient of a term T_j is "1", for instance, $T_3 = 1\cdot(x_2 \oplus 1)(x_3 \oplus 0)$, then the polarity of each literal in this term has a fixed state, eit-

her positive or negative but cannot take an arbitary one. If the coefficient of term T_j is "0", for instance, $T_5 = 0 \cdot (x_1 \oplus -)(x_2 \oplus -)$, then the polarity of each literal in this term can be arbitary, either negative or positive. We use "-" to denote the polarity of a literal when it can take an arbitary state.

Definition 3.5: In a complete GRM expansion f_e of an n-variable function, there are 2^{n-1} literals of any variable x_i. We define the collection of the 2^{n-1} polarities of these literals as the *Polarity Set of Variable* x_i. We denote this polarity set of variable x_i as $\delta_{f_e}(\dot{x}_i)$.

Definition 3.6: Let T be a product term of n distinct literals $T = \dot{x}_1 \dot{x}_2 \cdots \dot{x}_i \cdots \dot{x}_n$. The collection of the n polarities of these n literals is defined as the *Polarity Set of term* T. We use δ_T to represent this polarity set.

Definition 3.7: A collection of the n polarity sets of the n variables of a GRM expansion f_e is defined as *the Polarity Set of Expansion* f_e. This is denoted as $\delta_{f_e}(\dot{x}_1, \dot{x}_2, \cdots, \dot{x}_n)$.

The Polarity of a GRM expansion can also be represented by the termwise polarity set. For an n-variable function, the termwise polarity set is

$$\{\delta_{T_1}, \cdots \delta_{T_j}, \cdots \delta_{T_{2^n-1}}\}$$

Please note that for any GRM form, term T_0 is a constant $T_0 = b_0 \in \{0,1\}$, so for T_0 no polarity set exists.

Definition 3.8: A collection of the 2^n coefficients of a GRM expansion f_e is defined as *the coefficient set* of this expansion. Denoted by $f_e(b_0, b_1, \cdots, b_{2^n-1})$ this

coefficient set of a binary vector of length 2^n represents a GRM form under the selected polarities of their corresponding subsets of variables in products.

By the definitions above, we see that any GRM form of a function f can be identified by a polarity set and its corresponding coefficient set. The polarity set can be defined either termwise or variablewise.

Thus, the coefficient set of the expansion in example 3.1 is

$$f_e(00110001)$$

The polarity set of $\dot{x}_3$ is $\delta(x_3) = (-0-0)$.

The polarity set of $\dot{x}_2$ is $\delta(x_2) = (0\,1-1)$.

The polarity set of $\dot{x}_1$ is $\delta(x_1) = (---1)$.

The polarity set of term T_7 is $\delta_{T_7} = (110)$.

The polarity set of term T_6 is $\delta_{T_6} = (--)$.

The variablewise polarity set of the expansion is

$$\delta(\dot{x}_1,\dot{x}_2,\dot{x}_3) = (-0-0)(0\,1-1)(---1).$$

The termwise polarity set the expansion is

$$\{\delta_{T_1}, \delta_{T_2}, \delta_{T_3}, \delta_{T_4}, \delta_{T_5}, \delta_{T_6}, \delta_{T_7}\}$$
$$= \{(-), (0), (10), (-), (--), (--), (110)\}$$

Since in a FPRM expansion, the literals of a variable x_i can only take either positive or negative polarity but can not take both in the same expansion, only one polarity

it is necessary to denote the polarities of these 2^{n-1} literals of variable x_i. The polarity set of a FPRM expansion can be denoted as:

$$\delta_{\dot{x}_1\dot{x}_2\cdots\dot{x}_n}$$

or simply as a polarity vector δ_{f_e}

Example 3.2: The polarity of FPRM expansion

$$f = f_e = \bar{x}_2 \oplus \bar{x}_2 x_3 \oplus \bar{x}_1 \bar{x}_2 x_3$$

is written as $\delta_{\bar{x}_1\bar{x}_2x_3}$ or $\delta_{f_e}(\dot{x}_1\dot{x}_2\dot{x}_3) = (110)$

Definition 3.9: Adjacent Polarity GRM Expansions: f_{e_1} and f_{e_2} are two GRM expansions of the same function. If there is only one different bit between the polarity sets of these two expansions, we say that f_{e1} and f_{e2} are *Adjacent Polarity GRM Expansions*.

Example 3.3: Following are two GRM expansions of the same function f

$$f = f'_e = b'_0 \oplus b'_1\bar{x}_3 \oplus b'_2x_2 \oplus b'_3x_2x_3 \oplus b'_4x_1 \oplus b'_5x_1x_3 \oplus b'_6\bar{x}_1x_2 \oplus b'_7\bar{x}_1\bar{x}_2\bar{x}_3$$

and

$$f = f''_e = b''_0 \oplus b''_1\bar{x}_3 \oplus b''_2x_2 \oplus b''_3x_2x_3 \oplus b''_4\bar{x}_1 \oplus b''_5x_1x_3 \oplus b''_6\bar{x}_1x_2 \oplus b''_7\bar{x}_1\bar{x}_2\bar{x}_3$$

The polarity sets of f'_e and f''_e are $\delta_{f'_e}(\dot{x}_1,\dot{x}_2,\dot{x}_3) = (0011)(0001)(1001)$ and $\delta_{f''_e}(\dot{x}_1,\dot{x}_2,\dot{x}_3) = (1011)(0001)(1001)$, respectively. Since there is only one different bit between $\delta_{f'_e}(\dot{x}_1,\dot{x}_2,\dot{x}_3)$ and $\delta_{f''_e}(\dot{x}_1,\dot{x}_2,\dot{x}_3)$, f'_e and f''_e are adjacent polarity GRM

expansions.

Combining Definition 3.2 and Definiton 3.9 we obtain the definition of the Adjacent Polarity FPRM Expansions.

Definition 3.10: f_{e_1} and f_{e_2} are two FPRM expansions of the same function. If there is only one different bit between the polarity sets of f_{e_1} and f_{e_2}, we define that f_{e_1} and f_{e_2} are *Adjacent Polarity FPRM Expansions.*

Example 3.4: Following are two FPRM expansions of the same function f

$$f = f'_e = b'_0 \oplus b'_1 \bar{x}_3 \oplus b'_2 x_2 \oplus b'_3 x_2 \bar{x}_3 \oplus b'_4 \bar{x}_1 \oplus b'_5 \bar{x}_1 \bar{x}_3 \oplus b'_6 \bar{x}_1 x_2 \oplus b'_7 \bar{x}_1 x_2 \bar{x}_3$$

and

$$f = f''_e = b''_0 \oplus b''_1 x_3 \oplus b''_2 x_2 \oplus b''_3 x_2 x_3 \oplus b''_4 \bar{x}_1 \oplus b''_5 \bar{x}_1 x_3 \oplus b''_6 \bar{x}_1 x_2 \oplus b''_7 \bar{x}_1 x_2 x_3$$

The polarity sets of f'_e and f''_e are $\delta_{f'_e}(\dot{x}_1, \dot{x}_2, \dot{x}_3) = (101)$ and $\delta_{f''_e}(\dot{x}_1, \dot{x}_2, \dot{x}_3) = (100)$, respectively. Since there is only one different bit between $\delta_{f'_e}(\dot{x}_1, \dot{x}_2, \dot{x}_3)$ and $\delta_{f''_e}(\dot{x}_1, \dot{x}_2, \dot{x}_3)$, f'_e and f''_e are adjacent polarity FPRM expansions.

Theorem 3.1: For a Boolean function of n variables there exist $2^{E(n)}$ various GRM forms, where

$$E(n) = \sum_{i=1}^{n} i \begin{Bmatrix} n \\ i \end{Bmatrix}$$

Proof: Consider all possible GRM forms for a Boolean function of n-variables. There exist $\begin{Bmatrix} n \\ i \end{Bmatrix}$ subsets of variables with i variables in a subset. For each such subset there

exist two polarities of each variable, which means 2^i polarities for all the variables of the subset. Therefore the number of all possible GRM forms is

$$\prod_{i=1}^{n}(2^i)^{\binom{n}{i}} = \prod_{i=1}^{n} 2^{i\binom{n}{i}} = 2^{\sum_{i=1}^{n} i\binom{n}{i}} = 2^{E(n)}$$

It can be proven by mathematical induction that $E(n) = n\,2^{n-1}$.

It has been proved in [22] that any Boolean function of n variables($n \geq 3$) can be described by at most $\frac{3}{4}(2^{n-1})$ terms in a GRM form. This is the termwise upper bound of any function in a GRM form. Furthermore, for any function f (given in an RM form), there is an algorithm to find the above GRM form which takes $\frac{3}{4}(2^{n-1})$ steps.

III.3 The Problem of Minimization

The objective of logic minimization is to reduce the cost function associated with a logic expression. Our primary goal of this thesis is to minimize the number of terms in GRM expressions. The cost function C to be used here is :

$$cost = NT$$

Here NT is the total number of terms in the solution. According to our definition of GRM expression,

$$NT = \sum_{j=0}^{2^n-1} b_j$$

where b_j is the coefficient of term T_j.

CHAPTER IV

PARTIALLY-MIXED-POLARITY REED-MULLER EXPANSION

IV.1 Definitions

As stated in the previous chapter, the GRM forms are set up by selecting any combination of literals in the positive polarity(or zero polarity) RM expansion and replacing these with their inverses. They can be expressed as follows:

$$f(x_1,x_2,\cdots,x_i,\cdots,x_n) = b_0\dot{x}_1^0\dot{x}_2^0\cdots\dot{x}_i^0\cdots\dot{x}_n^0 \oplus b_1\dot{x}_1^0\dot{x}_2^0\cdots\dot{x}_i^0\cdots\dot{x}_n^1 \oplus \cdots \oplus$$
$$b_j\dot{x}_1^{e_1}\dot{x}_2^{e_2}\cdots\dot{x}_i^{e_i}\ldots\dot{x}_n^{e_n} \oplus \cdots \oplus b_{2^n-1}\dot{x}_1^1\dot{x}_2^1\cdots\dot{x}_i^1\cdots\dot{x}_n^1 \quad (4.1)$$

where $\dot{x}_i^0 = 1$, $\dot{x}_i^1 = \dot{x}_i$,(i=1,2,$\cdots$,n) ; $b_j \in \{0,1\}$, (j=0,1,2,$\cdots$, 2^n-1), $e_i \in \{0,1\}$, $(j)_{10}=(e_1e_2\cdots e_i\cdots e_n)_2$, and moreover, for each $\dot{x}_i$ in different terms we have $\dot{x}_i=x_i$ or $\dot{x}_i=\overline{x}_i$.

By putting some constraints on the GRM definition, the Partially-Mixed-Polarity Reed-Muller(PMPRM) expansion is established.

Definition 4.1: The Partially-Mixed-Polarity Reed-Muller(PMPRM) form is created by selecting any combination of the 2^{n-1} literals of one variable in the Fixed-Polarity RM expansion and replacing them with their inverses but keeping all the literals of the other variables under consistent fixed polarities.

Obviously, from the definition, the PMPRM forms are included in the GRM

forms but include the FPRM forms.

Let's consider a 3-variable function. It can be written as the following GRM expression

$$f = b_0 \oplus b_1(x_3 \oplus \gamma_1) \oplus b_2(x_2 \oplus \beta_1) \oplus b_3(x_3 \oplus \gamma_2) \oplus b_4(x_1 \oplus \alpha_1) \oplus b_5(x_1 \oplus \alpha_2)(x_3 \oplus \gamma_3)$$
$$\oplus b_6(x_1 \oplus \alpha_3)(x_2 \oplus \beta_3) \oplus b_7(x_1 \oplus \alpha_4)(x_2 \oplus \beta_4)(x_3 \oplus \gamma_4) \qquad (4.2)$$

where $\delta(\dot{x}_1) = (\alpha_1, \alpha_2, \alpha_3, \alpha_4)$, $\delta(\dot{x}_2) = (\beta_1, \beta_2, \beta_3, \beta_4)$ and $\delta(\dot{x}_3) = (\gamma_1, \gamma_2, \gamma_3, \gamma_4)$ are the polarity sets of each variable, respectively. Hence, there are $2^{3\cdot(2^3-1)} = 2^{12}$ GRM forms for a 3-variable function. For the PMPRM expansion, the literals of one variable can take mixed-polarities, and all the other variables should take fixed-polarities. If x_3 is the mixed-polarity variable, then x_1, x_2 must take fixed polarities. Thus, we have $\gamma_1, \gamma_2, \gamma_3, \gamma_4, \in \{0,1\}$ but $\alpha_1 = \alpha_2 = \alpha_3 = \alpha_4 \ (= \alpha) \in \{0,1\}$, and $\beta_1 = \beta_2 = \beta_3 = \beta_4$ $(= \beta) \in \{0,1\}$. Since there are 2^4 mixed polarities for the 4 literals of the mixed-polarity variable, 2^2 fixed polarities for the remaining two fixed polarity variables, and 3 choices of selecting the mixed polarity variable, thus there are 176(i.e. $3\cdot 2^2\cdot 2^4 - 2\cdot 2^3$) PMPRM forms for a 3-variable function. (The $3\cdot 2^2\cdot 2^4$ forms include 3 overlapping 2^3 fixed polarity forms, that is, all the three variables take fixed polarities, so $2\cdot 2^3$ should be subtracted).

Lemma 4.1: For an n-variable function, there are $n\,2^{n-1}2^{2^{n-1}} - (n-1)2^n$ PMPRM forms.

Proof: Each variable has 2^{n-1} literals in 2^n terms of the positive RM canonical form for an n-variable function. When the literals of one variable take the mixed polarities, there are $2^{2^{n-1}}$ mixed polarities for the variable. The other (n-1) variables should take the fixed polarities, and thus there are 2^{n-1} fixed polarities. For an n-variable func-

tion, we have n choices of selecting the mixed polarity variable, and therefore, we have $n\,2^{n-1}2^{2^{n-1}}$ alternative forms including the n-times overlapping 2^n FPRM forms. Hence, the total number of PMPRM forms is $n\,2^{n-1}2^{2^{n-1}} - (n-1)2^n$.

The relations among the PMPRM and the other RM expansion families are shown in Figure 2.

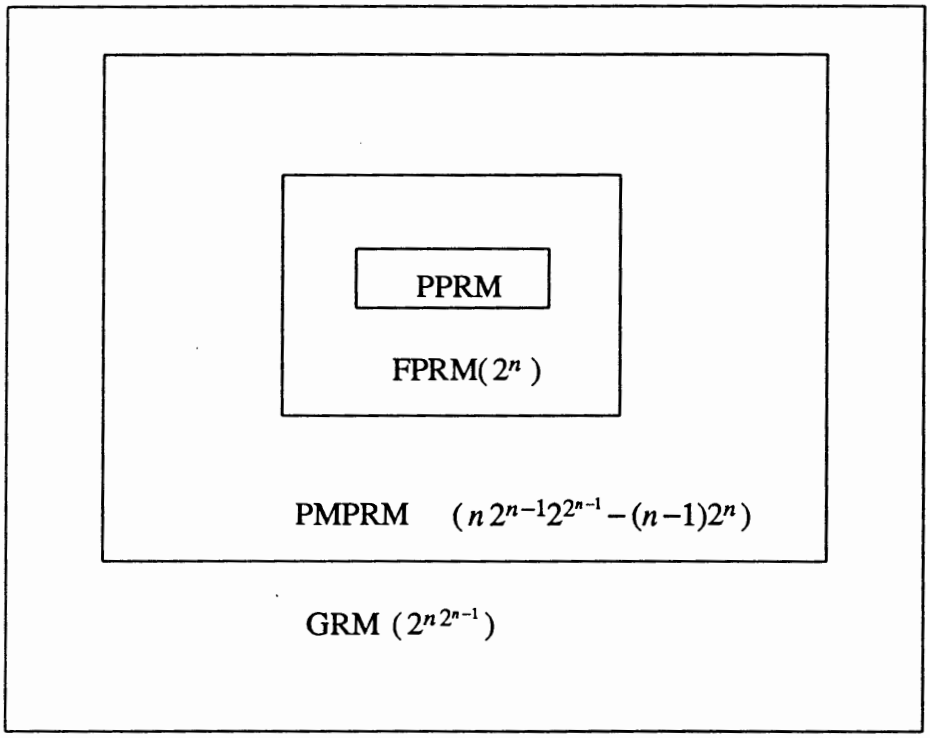


Figure 2. The Relationship among the PMPRM, PPRM, FPRM and GRM

By comparing Figure 2 and Figure 1, we see that the PMPRM class is much larger than the Kronecker Reed-Muller(KRM) expansion, thus generally the minimal form of this expansion will be much closer to the minimal ESOP than the minimal form of KRM expansion.

IV.2 Minimization of FPRM

An algorithm for the minimization of Fixed Polarity Reed-Muller expression has been presented by [18]. According to this algorithm, only 2^{n-1} Exor operations are needed to calculate the coefficients of FPRM expansion from one of its adjacent polarity expansions.

For instance, two FPRM adjacent polarity expansions of a 3-variable function are shown as the following:

$$f(x_1,x_2,x_3)=b_0\oplus b_1x_3\oplus b_2x_2\oplus b_3x_2x_3\oplus b_4x_1\oplus b_5x_1x_3\oplus b_6x_1x_2\oplus b_7x_1x_2x_3 \qquad (4.3)$$

$$f(x_1,x_2,x_3)=b'_0\oplus b'_1x_3\oplus b'_2\bar{x}_2\oplus b'_3\bar{x}_2x_3\oplus b'_4x_1\oplus b'_5x_1x_3\oplus b'_6x_1\bar{x}_2\oplus b'_7x_1\bar{x}_2x_3 \qquad (4.4)$$

Notice that between the FPRM expansions (4.3) and (4.4), there is only one variable x_2 whose polarity is inversed. To calculate the coefficient of Eqn.(4.4) from Eqn.(4.3), we substitute $\bar{x}_2$ with $1\oplus x_2$ in Eqn.(4.3) and obtain

$$\begin{aligned} f &= b_0\oplus b_1x_3\oplus b_2(\bar{x}_2\oplus 1)\oplus b_3(\bar{x}_2\oplus 1)x_3\oplus b_4x_1\oplus b_5x_1x_3\oplus b_6x_1(\bar{x}_2\oplus 1)\oplus b_7x_1(\bar{x}_2\oplus 1)x_3 \\ &= (b_0\oplus b_2)\oplus(b_1\oplus b_3)x_3\oplus b_2x_2\oplus b_3x_2x_3\oplus(b_4\oplus b_6)x_1\oplus(b_5\oplus b_7)x_1x_3\oplus b_6x_1x_2\oplus b_7x_1x_2x_3 \\ &= b'_0\oplus b'_1x_3\oplus b'_2\bar{x}_2\oplus b'_3\bar{x}_2x_3\oplus b'_4x_1\oplus b'_5x_1x_3\oplus b'_6x_1\bar{x}_2\oplus b'_7x_1\bar{x}_2x_3 \end{aligned}$$

where

$$b'_0=b_0\oplus b_2 \qquad b'_2=b_2$$

$$b'_1=b_1\oplus b_3 \qquad b'_3=b_3$$

$$b'_4=b_4\oplus b_6 \qquad b'_6=b_6$$

$$b'_5=b_5\oplus b_7 \qquad b'_7=b_7$$

This can be demonstrated by Figure 3:

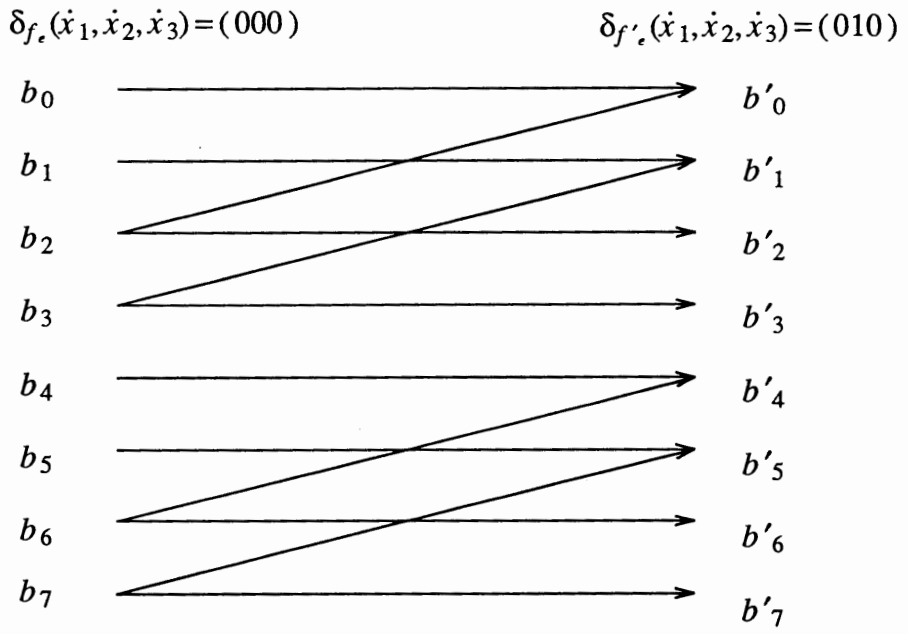


Figure 3. Calculation of f'_e from f_e

For an n-variable function, if all the polarities of the n variables are arranged according to the Gray code, then each polarity is adjacent to the next one, since the Gray code is a cyclic code, that is, in changing from one value to the next value only one bit is changed. Therefore, all 2^n sets of the RM polynomial coefficients may be arranged as the adjacent polarity based on Gray code ordering. Then the minimum coefficient can be obtained by exhaustive search through the adjacent polarity data flow chart.

Figure 4 is the flow graph representing the algorithm for the calculation of all FPRM expansions for a 3-variable function. Let us observe that in Figure 4 there are 8 different columns. Each represents one of the 2^3 fixed polarity expansions. Thus, seaching through all FPRM expansions leads to the exact minimum FPRM solution.

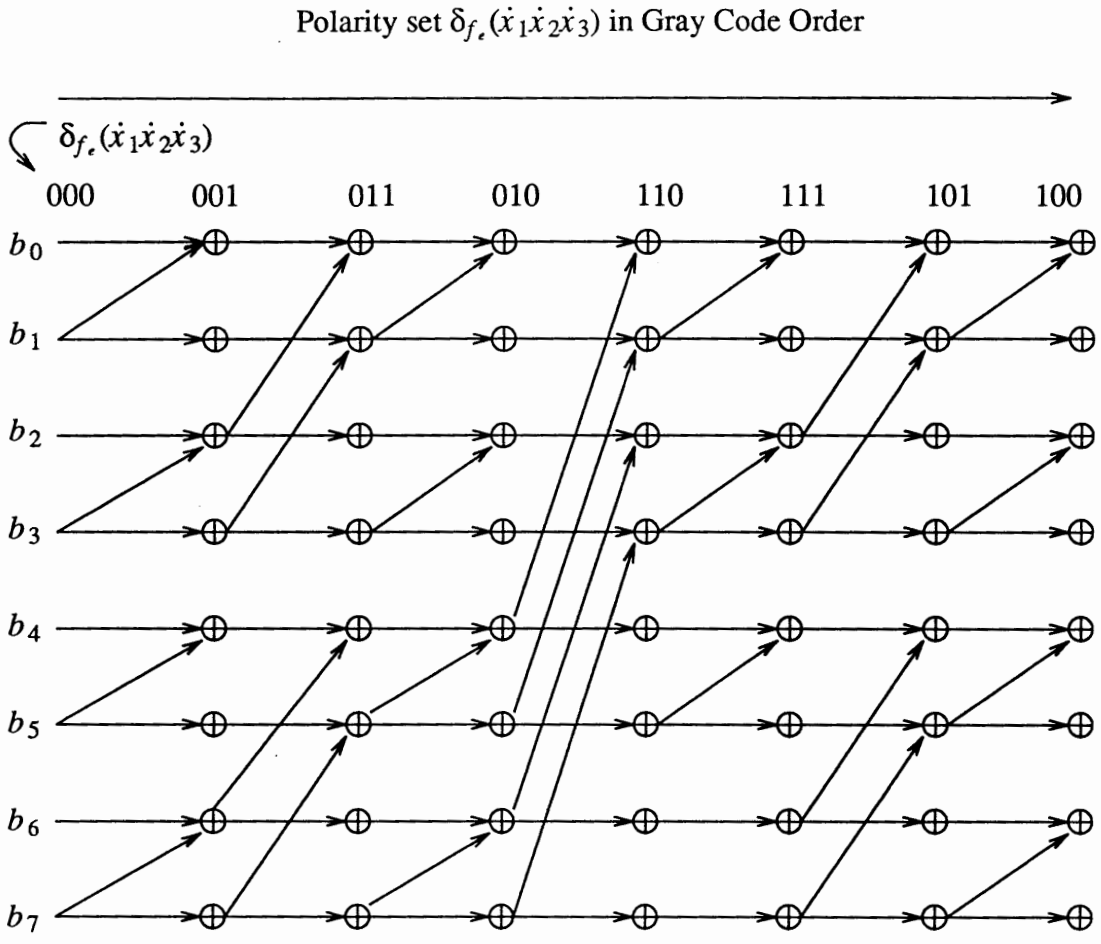


Figure 4. Calculation of FPRM expansions for a 3-variable Function

IV.3 Computation of the PMPRM Expansion

Definition 4.2: In PMPRM forms, if a mixed polarity of the 2^{n-1} literals of a variable has only one inverse bit with the other mixed polarity of the 2^{n-1} literals of the variable, then these two mixed polarities are defined as the Adjacent Polarities.

Theorem 4.1: When an n-variable PMPRM form shown in Eqn 4.1 is transformed to its adjacent polarity form, that is, a literal $\dot{x}_i$ in the term of $\dot{x}_1^{e_1}\dot{x}_2^{e_2}\cdots\dot{x}_i\cdots\dot{x}_n^{e_n}$ is inverted, only the $b_{j-2^{n-i}}$ is changed into $b_{j-2^{n-i}}\oplus b_j$, and all other coefficients remain unchanged.

Proof: In the form of Eqn 4.1, when the literal $\dot{x}_i$ in the term $\dot{x}_1^{e_1}\dot{x}_2^{e_2}\cdots\dot{x}_i^{e_i}\cdots\dot{x}_n^{e_n}(e_i=1)$ is inverted, we obtain its adjacent polarity form:

$$\begin{aligned} f(x_1,\cdots,x_i,\cdots,x_n)=&b'_0\dot{x}_1^0\dot{x}_2^0\cdots\dot{x}_i^0\cdots\dot{x}_n^0\oplus b'_1\dot{x}_1^0\dot{x}_2^0\cdots\dot{x}_i^0\cdots\dot{x}_n^1\oplus\cdots\oplus\\ &b'_{j-2^{n-i}}\dot{x}_1^{e_1}\dot{x}_2^{e_2}\cdots\dot{x}_i^0\cdots\dot{x}_n^{e_n}\oplus\cdots\oplus b'_j\dot{x}_1^{e_1}\dot{x}_2^{e_2}\cdots(\dot{x}_i\oplus 1)\cdots\dot{x}_n^{e_n}\\ &\oplus\cdots\oplus b'_{2^n-1}\dot{x}_1^1\dot{x}_2^1\cdots\dot{x}_i^1\cdots\dot{x}_n^1 \end{aligned} \tag{4.5}$$

From Eqn. 4.5, we have

$$\begin{aligned} f(x_1,\cdots,x_i,\cdots,x_n)=&b'_0\dot{x}_1^0\dot{x}_2^0\cdots\dot{x}_i^0\cdots\dot{x}_n^0\oplus b'_1\dot{x}_1^0\dot{x}_2^0\cdots\dot{x}_i^0\cdots\dot{x}_n^1\oplus\cdots\oplus\\ &b'_{j-2^{n-i}}\dot{x}_1^{e_1}\dot{x}_2^{e_2}\cdots\dot{x}_i^0\cdots\dot{x}_n^{e_n}\oplus b'_j\dot{x}_1^{e_1}\dot{x}_2^{e_2}\cdots\dot{x}_i^0\cdots\dot{x}_n^{e_n}\oplus\cdots\oplus\\ &b'_j\dot{x}_1^{e_1}\dot{x}_2^{e_2}\cdots\dot{x}_i\cdots\dot{x}_n^{e_n}\oplus\cdots\oplus b'_{2n-1}\dot{x}_1^1\dot{x}_2^1\cdots\dot{x}_i^1\cdots\dot{x}_n^1 \end{aligned} \tag{4.6}$$

By comparing Eqn. 4.6 to Eqn 4.1 we have

$$\begin{cases} b_{j-2^{n-i}}=b'_{j-2^{n-i}}\oplus b'_j\\ b_j=b'_j \end{cases} \tag{4.7}$$

Finally from Eqn. 4.7, we obtain

$$\begin{cases} b'_k=b_k & (k\neq j-2^{n-i})\\ b'_{j-2^{n-i}}=b_{j-2^{n-i}}\oplus b_j \end{cases} \tag{4.8}$$

From Theorem 4.1, if the mixed polarities of the 2^{n-1} literals of a variable are arranged in the Gray code order, each PMPRM form under the mixed polarities can be computed by using only one EXOR operation. The entry vectors are under the fixed polarities of the other variables and the zero -polarity of 2^{n-1} literals of this variable. The entry vectors are also computed in Gray code order of the fixed polarities[18]. Actually, this creates a two-dimensional data flow as shown in Figure 5(a). One dimension is

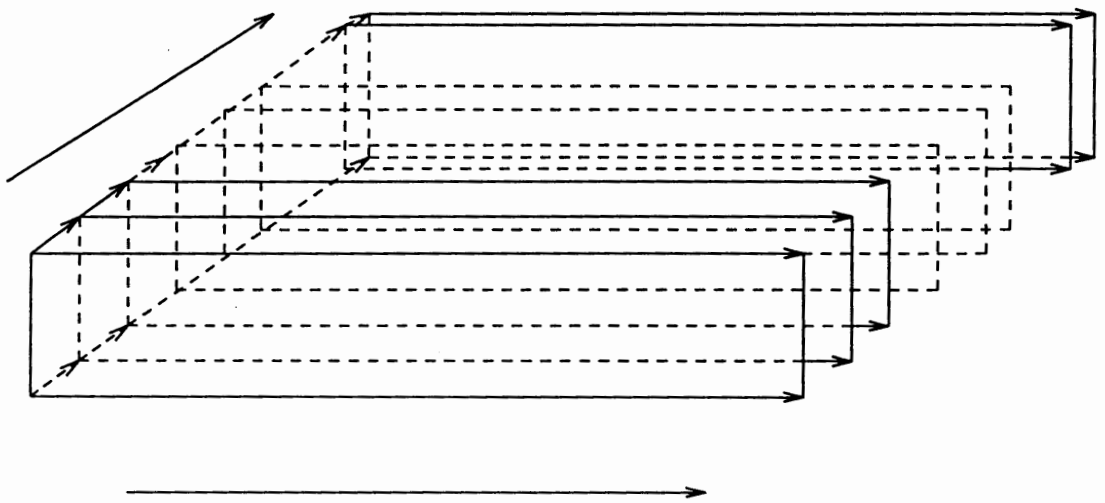


Figure 5(a). Two-dimensional data flow chart

in Gray code order of the fixed polarities while the other dimension is in Gray code order of the mixed polarities. Thus, according the Definition 4.2, all the PMPRM forms are generated.

In the following our fast computation algorithm will be explained by using a 3-variable function. In Eqn. 4.2, if the literals of variable x_3 take mixed polarities and x_1,x_2 take fixed polarities, one has $\alpha_1 = \alpha_2 = \alpha_3 = \alpha_4 (= \alpha) \in \{0,1\}$, $\beta_1 = \beta_2 = \beta_3 =$

$\beta_4 (=\beta) \in \{0,1\}$,and $\gamma_1, \gamma_2, \gamma_3, \gamma_4 \in \{0,1\}$. The expansions under mixed-polarities $\gamma_1, \gamma_2, \gamma_3, \gamma_4$ and fixed-polarities α, β is computed in the flow graph in Figure 6(a), where $\gamma_1, \gamma_2, \gamma_3, \gamma_4$ are arranged in Gray code order.

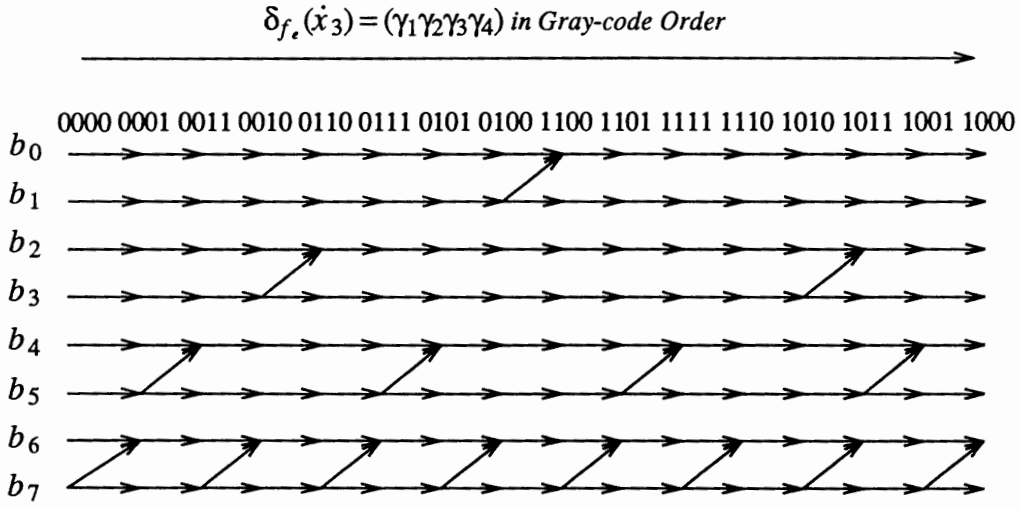


Figure 6(a) Flow Graph-Calculation of PMPRM forms when $\dot{x}_3$ is the mixed variable

In Figure 6(a) there are 16 columns, each represents one of the 2^4 PMPRM expansions under the mixed polarity of $\dot{x}_3$ and fixed polarity of $\dot{x}_1$ and $\dot{x}_2$. Throughout the data flow in Figure 6(a) α and β, the fixed polarities of $\dot{x}_1$ and $\dot{x}_2$, remain unchanged. The entry vector(first column) is under the zero(positive) polarity of $\dot{x}_3$ ($\delta_{f_e}(\dot{x}_3) = (\gamma_1\gamma_2\gamma_3\gamma_4) = (0000)$) and a certain states of α,β, the polarities of $\dot{x}_1$ and $\dot{x}_2$. Since $\alpha,\beta \in \{1,0\}$, so there are $2^2=4$ entry vectors with polarity set $\delta_{f_e}(\dot{x}_1,\dot{x}_2,\dot{x}_3) = (\alpha\beta 0)$. Each of these entry vector $[\dot{b}_1, \dot{b}_2, \cdots, \dot{b}_7]^T$ under fixed-polarities α, β in Fig. 6(a) is computed in the flow graph in Figure 6(b), where α, β are arranged in Gray code order.

Combining Figure 6(a) and Figure 6(b) results in a two-dimensional data flow graph, as shown in Figure 7.

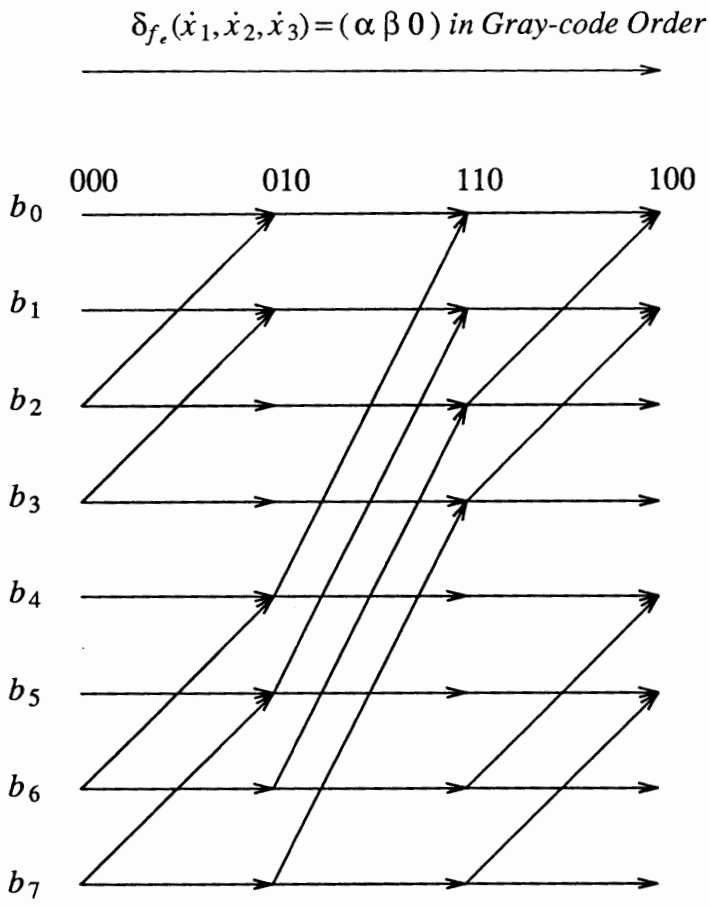


Figure 6(b) Calculation of FPRM forms: $\dot{x}_1$ and $\dot{x}_2$ be the fixed polarity variables

Similarly, if the literals of variable x_2 take mixed-polarities and x_1 , x_3 take fixed polarities, one has $\alpha_1=\alpha_2=\alpha_3=\alpha_4\ (=\alpha)\in\{0,1\}$, $\gamma_1=\gamma_2=\gamma_3=\gamma_4\in\{0,1\}$, and β_1 , β_2 , β_3 , $\beta_4\ (=\beta)\in\{\ 0,1\}$. The expansions under mixed-polarities β_1 , β_2 , β_3 , β_4 and fixed-polarities α , γ are computed in the flow graph in Figure 8(a), where β_1 , β_2 , β_3 , β_4 are arranged in Gray code order.

Each entry vector $[\dot{b}_1\ ,\dot{b}_2\ ,\ \cdots\ ,\dot{b}_7]^T$ under fixed-polarities α , γ in Fig. 8(a)is computed in the flow grap h in Figure 8(b), where α , γ are arranged in Gray code order.

If the literals of variable x_1 take mixed-polarities and x_2 , x_3 take fixed polarities, one has $\beta_1 = \beta_2 = \beta_3 = \beta_4 (= \beta) \in \{0,1\}$, $\gamma_1 = \gamma_2 = \gamma_3 = \gamma_4 \in \{0,1\}$, and α_1 , α_2 , α_3 , $\alpha_4 (= \alpha) \in \{0,1\}$. The expansions under mixed-polarities

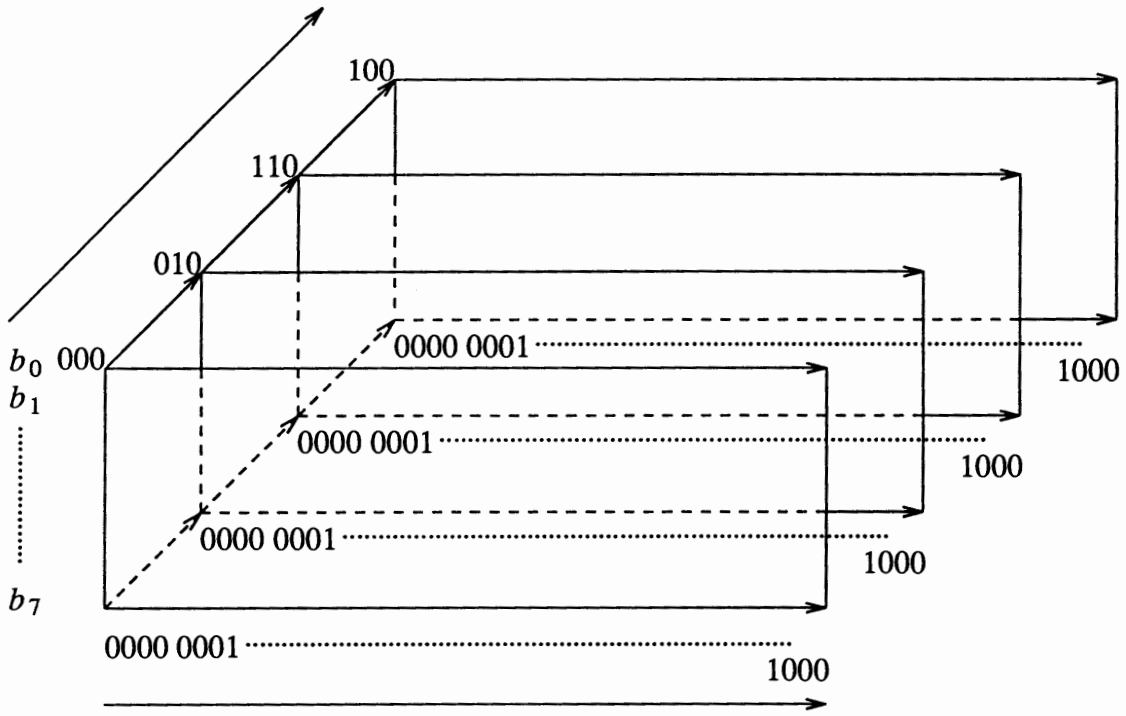


Figure 7 Two dimensional data flow graph with $\dot{x}_3$ being the mixed polarity variable

α_1 , α_2 , α_3 , $\alpha_4 (= \alpha) \in \{0 ,1\}$. and fixed polarities β , γ are computed in the flow graph in Figure 9(a) where α_1 , α_2 , α_3 , $\alpha_4 (= \alpha) \in \{0 ,1\}$ are arranged in Gray coder order. Each entry vector $[\dot{b}_1 , \dot{b}_2 , \cdots , \dot{b}_7]^T$ under fixed-polarities β , γ in Fig. 9(a) is computed in the flow graph in Figure 9(b), where β , γ are arranged in Gray code order.

As a result, the optimal polarity vector (one with the minimal number of nonzero elements) for the 3-variable function is selected among all the vectors.

Since the encoding of Gray codes is a reflective and cyclic encoding, for any n-variable function this algorithm is a recursive one, and thus it can be readily programmed. Since the algorithm excecutes only one EXOR operation for each mixed-polarity vector, this algorithm is highly efficient. The computation of PMPRM expansions for an n-variable function needs a total of $n(2^{n-1}-1)2^{n-1} + n\,2^{n-1}(2^{2^{n-1}}-1)$ EXOR operations.

The algorithm is formulated as follows:

Algorithm 4.1 (Minimization of PMPRM)

1. Start from PPRM, $f_e = PPRM$, $f_{\min} = f_e$
2. Let $i = 1$. Let mp be the polarity set of x_i and fp be the Fixed polarity vector of other variables.
3. $mp = mp + 1$ in Gray code order. Let T_j be the term in which the literal of x_i changes polarity. Now calculate the adjacent polarity PMPRM expansion f'_e from f_e by:

$$\begin{cases} b'_k = b_k & (k \neq j - 2^{n-i}) \\ b'_{j-2^{n-i}} = b_{j-2^{n-i}} \oplus b_j \end{cases}$$

4. If $cost(f'_e) < cost(f_{\min})$, then $f_{\min} = f'_e$
5. $f_e = f'_e$. If $mp < 2^{2^{n-1}}$, goto step 3 (search all possible mixed polarity expansions of x_i).
6. $fp = fp + 1$ in Gray code order. Calculate the adjacent polarity FPRM expansion under fp.
7. If $fp < 2^{n-1}-1$, go to step 3 (search all fixed-polarity expansions).

8. If $i < n$, then $i = i+1$, goto step 2 (go for all variables).

In this algorithm one has to calculate all possible PMPRM expansions including those overlaps. For functions with more than seven variables it is not feasible because the large number of forms. Figure 5(b) shows the search space for one mixed polarity variable of a n-variable function. It is essentially the same as Figure 5(a). To search all PMPRM expansions n numbers of Figure 5(b) are needed. Thus, the time complexity of this algorithm is $O(n \cdot 2^{n-1} 2^{2^{n-1}})$. In the next section we are going to give another algorithm which has the same result as algorithm 4.1 but with a time complexity of $O(n \cdot 2^n)$.

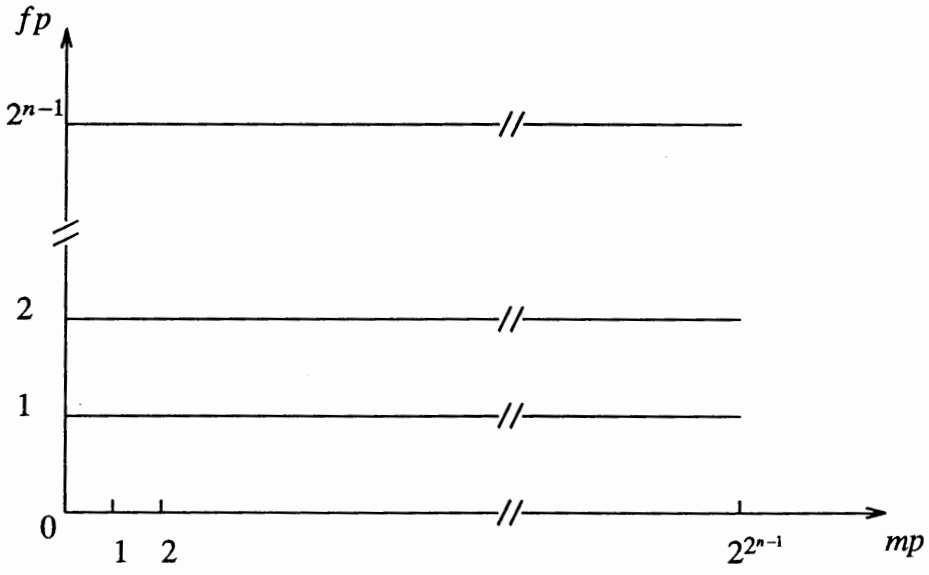
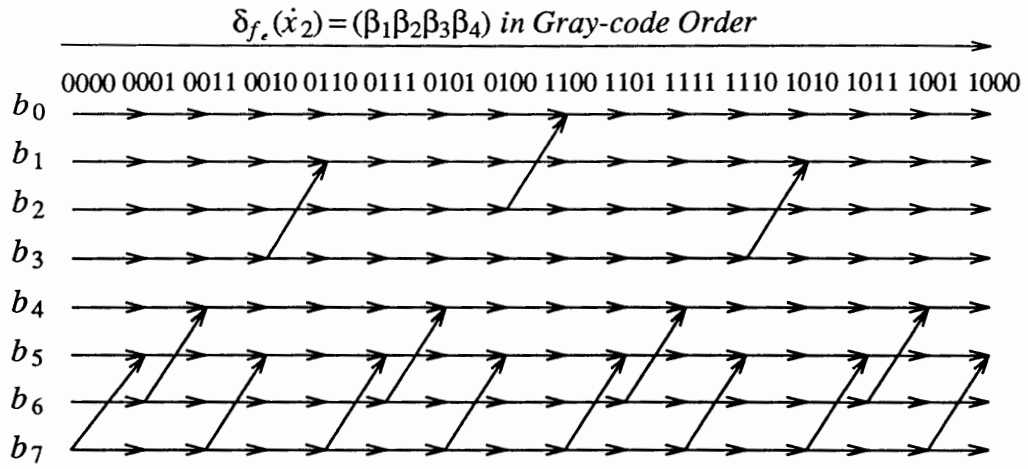
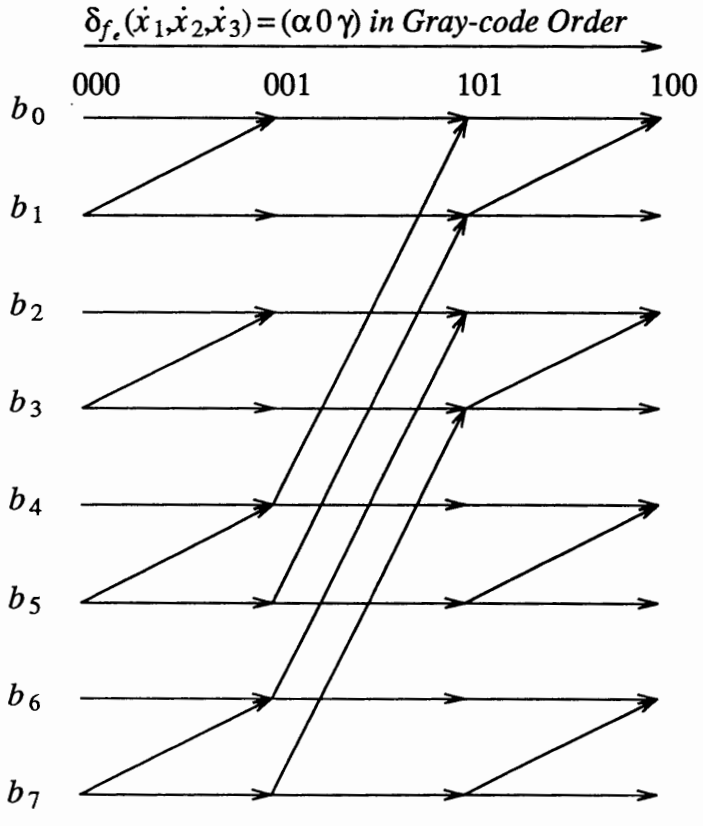


Figure5(b). Search Space of Algorithm 4.1 for One Mixed Polarity Variable

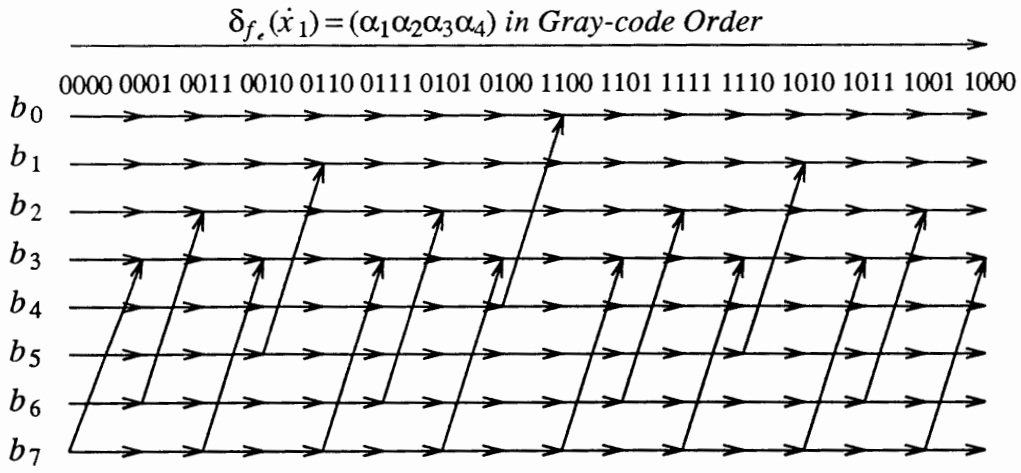


(a)

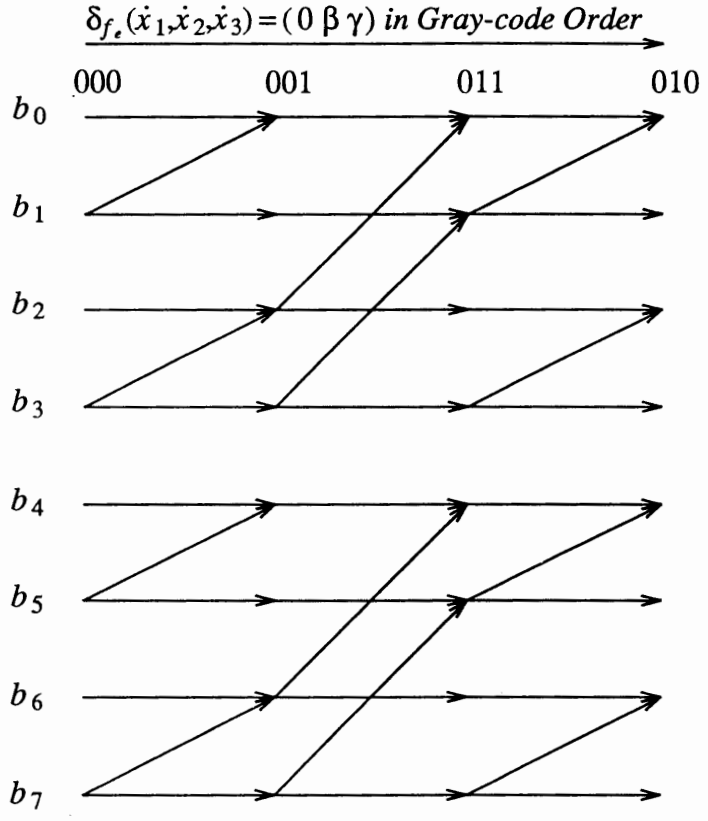


(b)

Figure 8. Flow Graph(a)($\beta_1\beta_2\beta_3\beta_4$) in Gray-code Order(b)($\alpha 0\gamma$) in Gray code Order



(a)



(b)

Figure 9. Flow Graph(a)($\alpha_1\alpha_2\alpha_3\alpha_4$)in Gray code order(b)($0\beta\gamma$) in Gray code order.

IV.4 The Speedup Approach

Using Algorithm 4.1, one has to calculate all the PMPRM forms in order to find the minimum one. This would not be feasible for functions that have many input variables. In this section, we introduce an algorithm with which we can find the minimum PMPRM forms without necessarily calculating all the forms. Before we present the algorithm formally, let us consider the following example first:

Example 4.1: Given is a function in PPRM form:

$$f = f_e = 1 \oplus x_2 \oplus x_2x_3 \oplus x_1x_3 \oplus x_1x_2 \oplus x_1x_2x_3$$
$$= 1 \oplus 0\cdot(x_3 \oplus \gamma_1) \oplus x_2 \oplus x_2(x_3 \oplus \gamma_2) \oplus x_1(x_3 \oplus \gamma_3) \oplus x_1x_2 \oplus x_1x_2(x_3 \oplus \gamma_4)$$

where $\delta_{f_e} = (\gamma_1\gamma_2\gamma_3\gamma_4) = (0000)$. By inverting all the literals of x_3 in the above expansion, we obtain a FPRM form as the following:

$$f = f'_e = 1 \oplus x_2\bar{x}_3 \oplus x_1 \oplus x_1\bar{x}_3 \oplus x_1x_2\bar{x}_3$$
$$= 1 \oplus 0\cdot(x_3 \oplus \bar{\gamma}_1) \oplus x_2(x_3 \oplus \bar{\gamma}_2) \oplus x_1 \oplus x_1(x_3 \oplus \bar{\gamma}_3) \oplus x_1x_2(x_3 \oplus \bar{\gamma}_4)$$

where $\delta_{f'_e} = (\bar{\gamma}_1\bar{\gamma}_2\bar{\gamma}_3\bar{\gamma}_4) = (1111)$. Comparing f_e and f'_e we find that the inversion of γ_1 does not cause any change because the coefficient of term $0\cdot(x_3 \oplus \gamma_1)$ is 0, thus γ_1 can be arbitary. The inversing of γ_2 (x_3 in term x_2x_3) in f_e causes the term x_2 to be reduced in f'_e. The inversing of γ_4 (x_3 in term $x_1x_2x_3$) causes term x_1x_2 to be reduced in f'_e. But the inversing of γ_3 (x_3 in term x_1x_3) creates one more term x_1 in f'_e. Now if we only invert polarities γ_2 and γ_4 but leave the polarities γ_1 and γ_3 unchanged, we have a PMPRM expansion as follows:

$$f = f''_e = 1 \oplus x_2\bar{x}_3 \oplus x_1x_3 \oplus x_1x_2\bar{x}_3$$
$$= 1 \oplus 0\cdot(x_3 \oplus \gamma_1) \oplus x_2(x_3 \oplus \bar{\gamma}_2) \oplus x_1(x_3 \oplus \gamma_3) \oplus x_1x_2(x_3 \oplus \bar{\gamma}_4)$$

This is the minimum PMPRM form with zero polarity of x_1, x_2 and mixed polarity of x_3.

From Theorem 4.1 and Figures 5(a), 6(a), and 7(a) we observe that whenever a literal $\dot{x}_i$ in term T_j is inverted, only the coefficient $b_{j-2^{n-i}}$ is changed to $b_{j-2^{n-i}} \oplus b_j$. This coefficient is only affected by the literal $\dot{x}_j$ in term T_j, no matter how the literals of the same variable in other terms are changed.

For $b'_{j-2^{n-i}} = b_{j-2^{n-i}} \oplus b_j$, the following combinations are possible if $\dot{x}_i$ in term T_j is inverted (Note this inversion can be from x_i to $\overline{x}_i$ or from $\overline{x}_i$ to x_i).

$b_{j-2^{n-i}}$	0	0	1	1
b_j	0	1	0	1
$b'_{j-2^{n-i}} = b_{j-2^{n-i}} \oplus b_j$	0	1	1	0

Table 1. Possible Combination of $b'_{j-2^{n-i}}$

From Table 1 we see only when $b_{j-2^{n-i}}$ and b_j both equal to "1" that the cost will be improved.

Owing to the above reasons, the minimum PMPRM under the mixed-polarity of x_i and a certain fixed polarities of other variables can be obtained by attempting to invert any single literal from the 2^{n-1} literals of x_i. If the inversion of literal $\dot{x}_i$ in term T_j causes $b_{j-2^{n-i}}$ to change from "1" to "0", then the inverted literal is the one that should stand in the minimum PMPRM. Otherwise the original literal will leave in the minimum PMPRM. Again we use a 3-variable function to explain our algorithm. In Eqn. 4.2, if the literals of variable x_3 take mixed polarities and x_1, x_2 take fixed polari-

ties, we first invert the polarities γ_1, γ_2, γ_3, and γ_4 and keep α_1, α_2, α_3, α_4 and β_1, β_2, β_3, β_4 unchanged. The resulting adjacent polarity FPRM expansion has a coefficient vector $f(b'_0,b'_1,b'_2,b'_3,b'_4,b'_5,b'_6,b'_7)$ and a polarity set $(\gamma'_1\gamma'_2\gamma'_3\gamma'_4)=(1111)$, as shown in Figure 10. Let $f(b''_0,b''_1,b''_2,b''_3,b''_4,b''_5,b''_6,b''_7)$ be the coefficient set of the minimum PMPRM, and $(\gamma''_1\gamma''_2\gamma''_3\gamma''_4)$ be the corresponding polarity set, then

$$\begin{aligned} &\text{if } b_j=1 \text{ and } b'_j=0 \qquad \text{then} \quad b''_j=b'_j, \quad \gamma''_k=\gamma'_k \\ &\text{for others} \qquad\qquad\qquad\qquad\quad b''_j=b_j, \quad \gamma''_k=\gamma_k \end{aligned}$$

In Example 4.1, $f(b_0,b_1,b_2,b_3,b_4,b_5,b_6,b_7)=f(10110111)$ and $(\gamma_1=\gamma_2=\gamma_3=\gamma_4)=(0000)$, we have $f(b'_0,b'_1,b'_2,b'_3,b'_4,b'_5,b'_6,b'_7)=f(10011101)$ with $(\gamma'_1=\gamma'_2=\gamma'_3=\gamma'_4)=(1111)$.

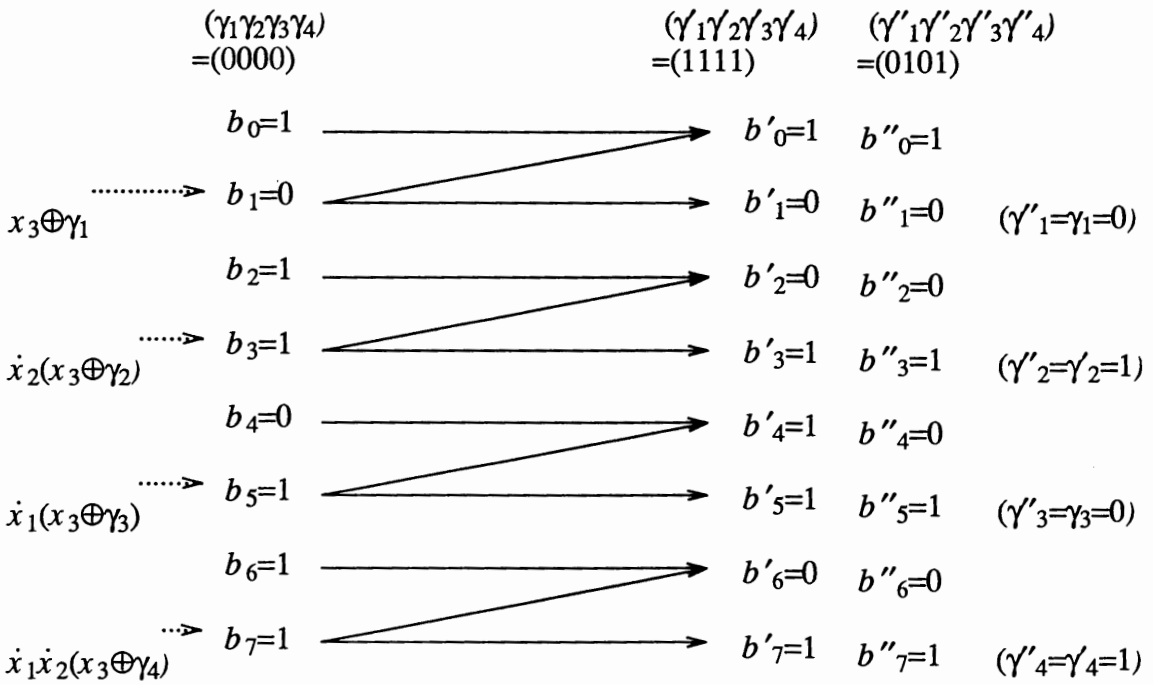


Figure 10. Calculation of the Minimal PMPRM for a 3-variable Function

Thus the minimum PMPRM will be

$f(b''_0,b''_1,b''_2,b''_3,b''_4,b''_5,b''_6,b''_7)=f(10010101)$ with the polarity set of x_3 $\delta(\dot{x}_3)=(\gamma'_1\gamma'_2\gamma'_3\gamma'_4)=(0101)$.

Figure 10 is actually the calculation of Adjacent Polarity FPRM Expansion under which x_3 is the variable that is inverted. From Figure 8 we obtain the same result as in Figure 6(a). By combining Figure 10 and Figure 6(b) we obtain Figure 11 from which we can obtain the minimum PMPRM with x_3 being the mixed polarity variable.

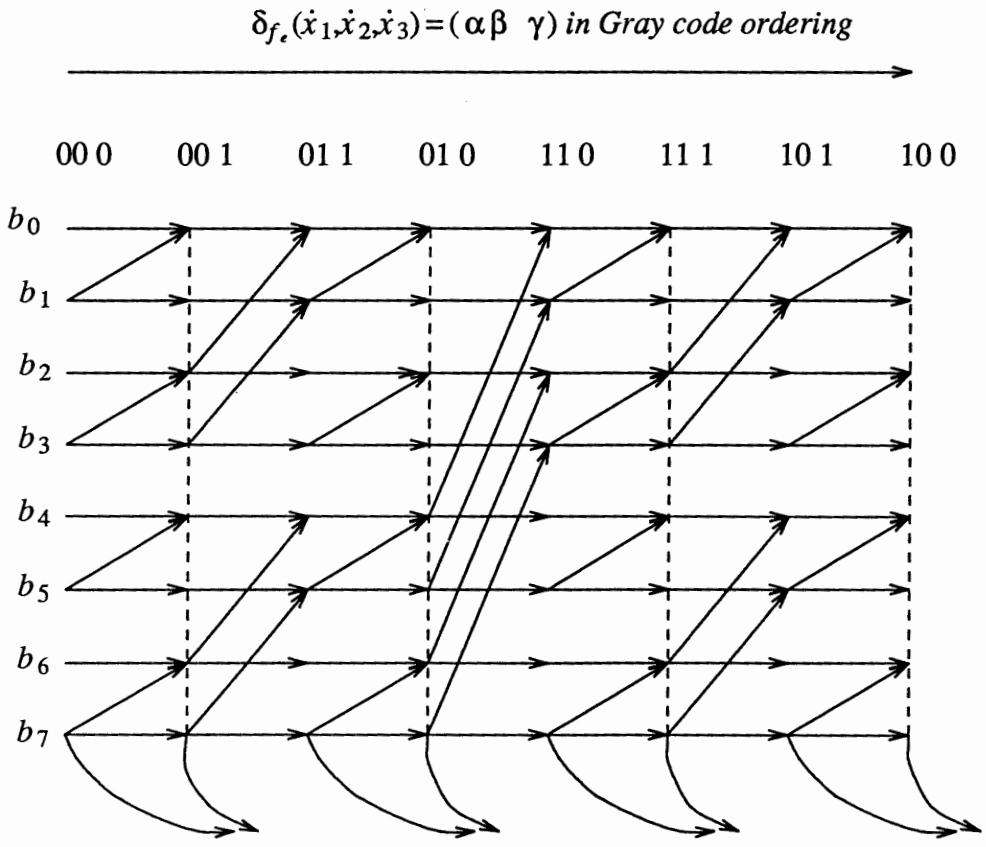


Figure 11. Flow graph, α,β,γ in Gray-code Order

In Figure 11, γ is arranged as the least significant bit (1st bit). β and α are arranged as the 2nd and the 3rd bits, respectively. During every increment in Gray-code ordering, if the inversed bit is the 1st bit, then we can calculate the optimal PMPRM expansion, as stated in the above. If the inversed bit is the 2nd or the 3rd, then we can calculate the FPRM under fixed polarities of α and β, the same result as from Figure 6(b). In order to distinguish the mixed polarity variable and the fixed polarity variables, in Figure 11 we leave a space between the least significant bit and the others.

Similarly, Figure 8 and Figure 9 can be transformed into data flow charts, as shown in Figure 12 and Figure 13, in which the polarity bit of the mixed polarity variable is always arranged as the least significant bit. The algorithm is presented as the following:

Algorithm 4.3: (Fast minimization of PMPRM)

1. Start from PPRM, let $f_{\min} = PPRM$. $cost\_min = cost(PPRM)$.
2. $i = 1$. Let x_i be the mixed polarity variable; $f_e = PPRM$; fp be the fixed polarity vector in which the polarity of x_i is arranged as the least significant bit. Let δ be the zero polarity of x_i. δ' be the polarity set of x_i in the minimal GPMPRM.
3. $fp = fp + 1$ in Gray code order. Calculate the Adjacent Polarity FPRM Expansion f'_e from f_e by:

$$
\begin{aligned}
&for(j=0;j<=2^n-1;j++)\{\\
&\quad if(T_{j+2^{n-i}} \text{ contains literal } x_i)\\
&\qquad b'_j = b_j \oplus b_{j+2^{n-i}};\\
&\quad else\\
&\qquad b'_j = b_j;\}
\end{aligned}
$$

4. If $cost(f'_e) < cost\_min$, then $f_{\min} = f'_e$.

5. If the bit changed is the least significant bit, derive the optimal PMPRM f''_e from f'_e and f_e by:

   if $b_j = 1$ and $b'_j = 0$ then $b''_j = b'_j$, $\gamma''_k = \gamma'_k$

   for others $b''_j = b_j$, $\gamma''_k = \gamma_k$

6. If $cost(f''_e) < cost\_min$, then $f_{min} = f''_e$.

7. $f_e = f'_e$. If $fp < 2^n$, Goto step 2 (search all mixed polarity expansions of $\dot{x}_i$).

8. If $i < n$, then $i = i + 1$. Goto step 2 (go through all variables).

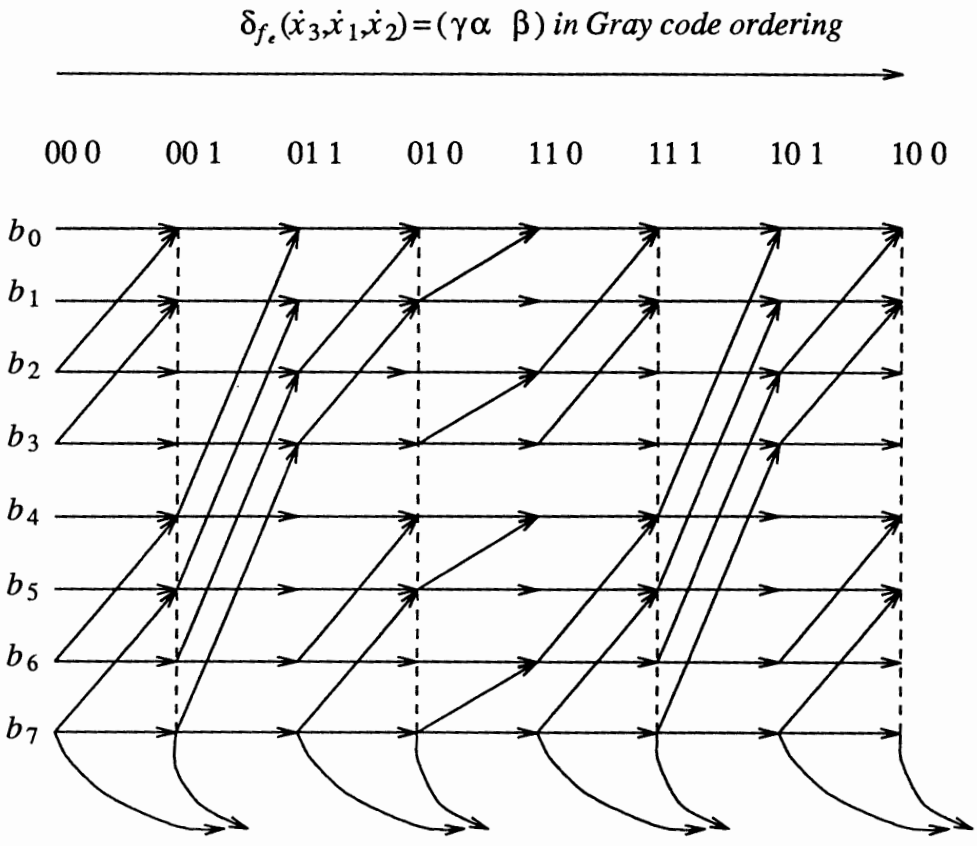


Figure 12. Flow Graph, γ,α,β in Gray-code Order

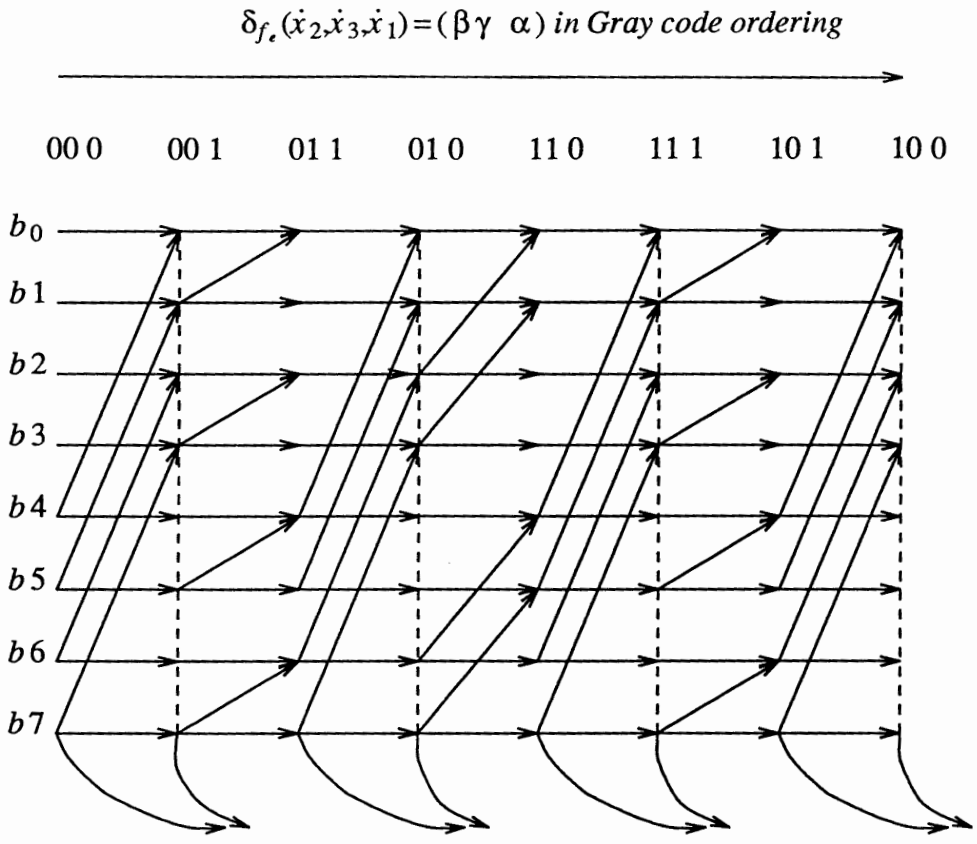


Figure 13. Flow graph, β,γ,α in Gray-code Order

IV.5 The Efficiency of Algorithm 4.2

The criteria used to evaluate the efficiency of an algorithm is the time and space complexities. From the previous section we can see that to find the minimum FPRM expansion one has to calculate 2^n FPRM forms. By calculating $n\,2^n$ FPRM forms, we can obtain the minimum PMPRM expansion, which is much closer to the minimum ESOP, than is the minimum FPRM form. The time complexity of algorithm 4.2 is $O(n \cdot 2^n)$. This time complexity can further be improved if parallel processing technique

is taken into consideration. The parallarism of Algorithm 4.2 is quite obvious. First, each pass of the calculations of the PMPRM expansions with a particular mixed polarity variable starts from the PPRM expansion, and all these passes are irrelative to each other. Second, the Gray code is reflective. This make it possible to start a pass from different directions at the same time. As an example, the minimization of a three variable function is demonstrated in Figure 14. The time complexity of Algorithm 4.2 using strategy of parallel processing is $O(2^{n-1})$. This improvement is achieved at the cost of more memory space.

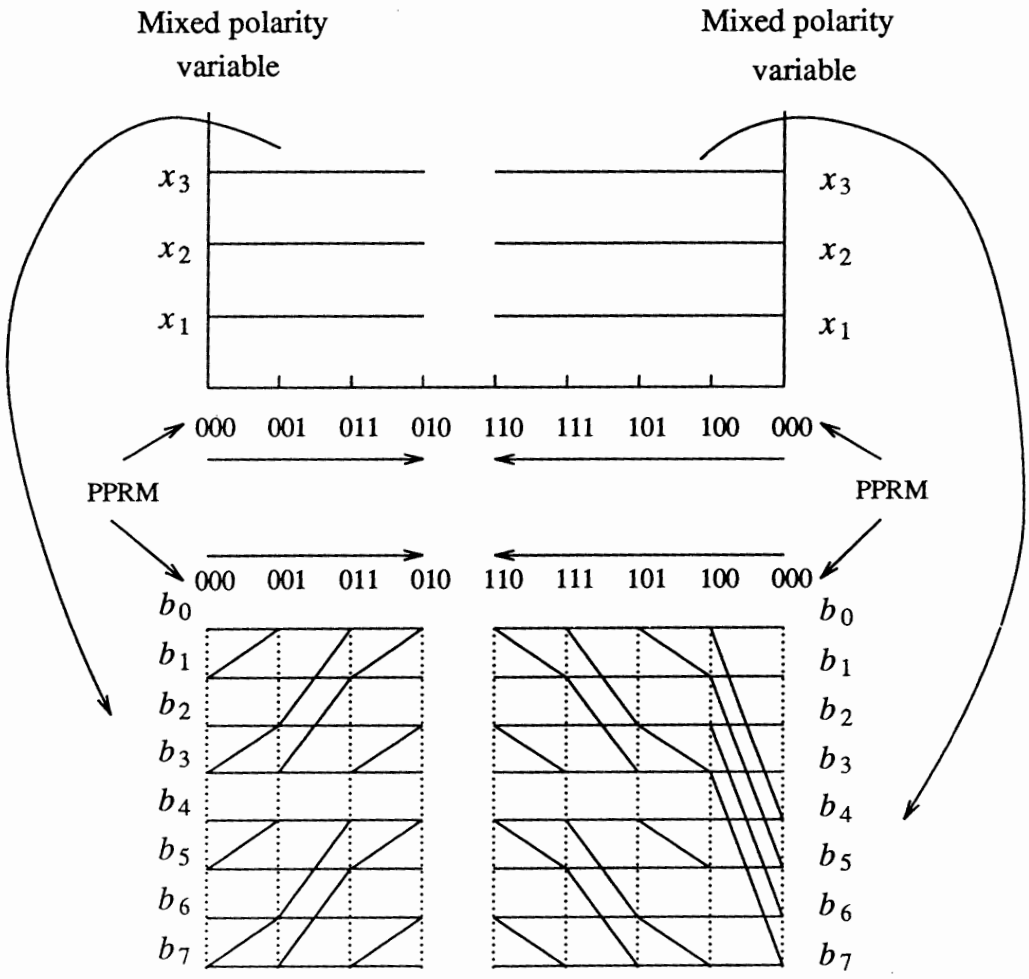


Figure 14. Minimization Algorithm Using Parallel Prosessing Technique

Because our algorithms works on the coefficients of the Reed-Muller expansion, all the termwise polarity sets of two expansions have to be keep throughout the program. One is the current expansion and the other is the expansion with best cost. There are a total of 2^n terms. Each term need a maximum of n memory spaces for the polarities of each literal. Thus the space complexity of this algorithm is $O(2 \cdot n \cdot 2^n) = O(n \cdot 2^{n+1})$.

CHAPTER V

MINIMIZATION OF THE GENERALIZED REED-MULLER EXPANSION

Similarly to the algorithm described in the previous chapter, if we can calculate the Adjacent Polarity GRM expansion one-by-one and search all the GRM forms the minimum can be found. But the computation of the Adjacent Polarity GRM expansion is not as simple as for the PMPRM's. Consider a simple example:

$$f = x_2 \oplus \bar{x}_2 x_3 \oplus \bar{x}_1 x_2 x_3$$

This is a GRM expression of a 3-variable function under the polarity of $\delta_f(\dot{x}_1\dot{x}_2\dot{x}_3) = (---1)(01-0)(-0-0)$. One of the adjacent polarity set would be $\delta_f(\dot{x}_1\dot{x}_2\dot{x}_3) = (---0)(01-0)(-0-1)$. To calculate the coefficients of this expansion, we substitute the literal $\bar{x}_1$ in term $\bar{x}_1 x_2 x_3$ in with $1 \oplus x_1$ and obtain:

$$\begin{aligned} f &= x_2 \oplus \bar{x}_2 x_3 \oplus ((x_1 \oplus 1) x_2 x_3) \\ &= x_2 \oplus \bar{x}_2 x_3 \oplus (x_2 x_3 \oplus x_1 x_2 x_3) \\ &= x_2 \oplus \bar{x}_2 x_3 \oplus ((\bar{x}_2 \oplus 1) x_3 \oplus x_1 x_2 x_3) \\ &= x_2 \oplus \bar{x}_2 x_3 \oplus (\bar{x}_2 x_3 \oplus x_3 \oplus x_1 x_2 x_3) \\ &= x_2 \oplus x_3 \oplus x_1 x_2 x_3 \end{aligned}$$

From the above, we see that $b'_3 = b_3 \oplus b_7 = 0$, $b'_1 = b_1 \oplus b_7 = 1$. In this example, the cost function(we consider the number of terms only) does not improve.

In this chapter we discuss the problem of how to minimize the GRM forms using

an algorithm similar to that described in chapter IV.

V.1 Termwise Complementary Expansion Diagram

Theorem 5.1 f_{1e} and f_{2e} are GRM forms of two different n-variable functions f_1 and f_2. The coefficient sets of f_{1e} and f_{2e} are $f_{1e}(b_{10}b_{11}b_{12}\cdots b_{12^n-1})$ and $f_{2e}(b_{20}b_{21}b_{22}\cdots b_{22^n-1})$, respectively. If f_{1e} and f_{2e} are with the same polarity, then $f_e = f_{1e} \oplus f_{2e}$ is the GRM form of function $f = f_1 \oplus f_2$ with the same polarity as f_{1e} and f_{2e}. The coefficient set of f_e is:

$$\begin{aligned} f_e(b_0b_1\cdots b_{2^n-1}) &= f_{1e}(b_{10}b_{11}\cdots b_{12^n-1}) \oplus f_{2e}(b_{20}b_{21}\cdots b_{22^n-1}) \\ &= f_e(b_{10}\oplus b_{20}, b_{11}\oplus b_{21}, \ldots, b_{12^n-1}\oplus b_{22^n-1}) \end{aligned}$$

also, let $f_{1e}, f_{2e}, \cdots$, and f_{me} be GRM forms of m different n-variable functions but all these GRM forms are under the same polarity. Then the GRM expression of function $f = f_1 \oplus f_2 \oplus \cdots \oplus f_m$ can be obtained by

$$\begin{aligned} f_e(b_0b_1\cdots b_{2^n-1}) &= f_{1e}(b_{10}b_{11}\cdots b_{12^n-1}) \oplus f_{2e}(b_{20}b_{21}\cdots b_{22^n-1}) \\ &\quad \oplus \cdots \oplus f_{me}(b_{m0}b_{m1}\cdots b_{m2^n-1}) \end{aligned}$$

where

$$\begin{aligned} b_0 &= b_{10}\oplus b_{20}\oplus \cdots \oplus b_{m0} \\ b_1 &= b_{11}\oplus b_{21}\oplus \cdots \oplus b_{m1} \\ b_2 &= b_{12}\oplus b_{22}\oplus \cdots \oplus b_{m2} \\ &\vdots \\ b_{2^n-1} &= b_{12^n-1}\oplus b_{22^n-1}\oplus \cdots \oplus b_{m2^n-1} \end{aligned}$$

Proof: Trivial.

Example 5.1

Given are two 3-variable functions in GRM forms.

$$f_1 = f_{1e} = 1 \oplus x_3 \oplus x_1 \oplus \bar{x}_1 x_2 \oplus x_1 \bar{x}_2 x_3$$
$$f_2 = f_{2e} = x_2 \oplus x_1 \oplus \bar{x}_1 x_3 \oplus \bar{x}_1 x_2$$

The polarity sets and coefficients sets of f_{1e} and f_{2e} are $\delta_{f_{1e}}(\dot{x}_1\dot{x}_2\dot{x}_3)$=(0–10)(– –01)(0—0), $\delta_{f_{2e}}(\dot{x}_1\dot{x}_2\dot{x}_3)$=(011–)(0–0–)(– –0–) and f_{1e}(11001011), f_{2e}(00101110), respectively. Since in the polarity set a "-" stands for either a "1" or a "0", the common polarity set of the above two expansions is $\delta_{f_e}(\dot{x}_1\dot{x}_2\dot{x}_3)$=(0110)(0–01)(0–00). From Theorem 5.1 we obtain the GRM expansion of function $f = f_1 \oplus f_2$ under the polarity of $\delta_{f_e}(\dot{x}_1\dot{x}_2\dot{x}_3)$=(0110)(0–01)(0–00)

$$f_e(b_0b_1b_2b_3b_4b_5b_6b_7) = f_{1e}(11001011) \oplus f_{2e}(00101110)$$
$$= f_e(11100101)$$

Definition 5.1: Let T be a product term of n distinct literals $T = \dot{x}_1\dot{x}_2 \cdots \dot{x}_i \cdots \dot{x}_n$. If we apply property $\dot{x} = 1 \oplus \bar{x}$ to all the literals in T, we have a FPRM expansion as the following:

$$T = \dot{x}_1\dot{x}_2 \cdots \dot{x}_i \cdots \dot{x}_n$$
$$= 1 \oplus \bar{x}_n \oplus \bar{x}_{n-1} \oplus \bar{x}_n\bar{x}_{n-1} \oplus \cdots \oplus \bar{x}_1\bar{x}_2 \cdots \bar{x}_n$$

The polarities of all the variables in the above expansion are the complements of the corresponding variables in product T. This expansion is defined as the *Complementary Expansion* of product T.

The complementary expansion of product T can be readily obtained because it is a FPRM expansion with all the coefficients equal to one.

Expample 5.2

The complementary expansion of product term $x_1x_2\bar{x}_3$ is

$$1\oplus x_3\oplus\bar{x}_2\oplus\bar{x}_2x_3\oplus\bar{x}_1\oplus\bar{x}_1x_3\oplus\bar{x}_1\bar{x}_2\oplus\bar{x}_1\bar{x}_2x_3$$

Definition 5.2: If we apply $\dot{x}=1\oplus\bar{x}$ only to a selected set of literals $\dot{x}_{k_1},\dot{x}_{k_2},\cdots,$ and $\dot{x}_{k_m}$ in term T ($T=\dot{x}_1\dot{x}_2\cdots\dot{x}_n$ and $k_1,k_2,\cdots,k_m \in \{1,2,\cdots,n\}$), we have a FPRM expansion. The polarities of the literals of $\dot{x}_{k_1},\dot{x}_{k_2},\cdots,\dot{x}_{k_m}$ are complements of the corresponding literals in product T. The polarities of other literals remain the same as those in product T. This expansion is called the *Complementary Expansion of T versus literals* $\dot{x}_{k_1},\dot{x}_{k_2},\cdots,$ *and* $\dot{x}_{k_m}$.

The complementary expansion of T versus literals $\dot{x}_{k_1},\dot{x}_{k_2},\cdots,\dot{x}_{k_m}$ can be obtained by first calculating the complementary expansion of $\dot{x}_{k_1}\dot{x}_{k_2}\cdots\dot{x}_{k_m}$ and then multiplying the other literals.

Example 5.3

The complementary expansion of term $x_1x_2\bar{x}_3$ versus $\bar{x}_3$ is

$$x_1x_2(1\oplus x_3)$$
$$=x_1x_2\oplus x_1x_2x_3$$

The complementary expansion of term $x_1x_2\bar{x}_3$ versus x_2 and $\bar{x}_3$ is

$$x_1 \cdot (1 \oplus \bar{x}_2 \oplus x_3 \oplus \bar{x}_2 x_3)$$
$$= x_1 \oplus x_1 \bar{x}_2 \oplus x_1 x_3 \oplus x_1 \bar{x}_2 x_3$$

The complementary expansion of T can be illustrated by a subtree where T is the root and the terms in the complementary expansion are represented by the descendents of T. For instance, the complementary expansion of $T = x_1 x_2 \bar{x}_3$ versus x_2 and $\bar{x}_3$ can be illustrated by the following diagram:

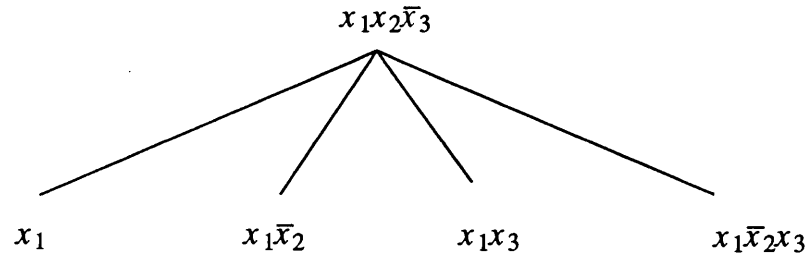


Figure 15. Complementary Expansion of $x_1 x_2 \bar{x}_3$ versus x_2 and $\bar{x}_3$

By the complementary expansion one can always transform a product term with any polarity into a FPRM expansion which includes a term $T' = \dot{x}_1 \dot{x}_2 \cdots \dot{x}_n$ with expected polarities. If we apply the complementary expansion to all the terms of a GRM form and to all the terms of the resulting complementary expansions, then a GRM form with the expected polarity can be obtained.

Example 5.4: Consider a 4-variable function

$$f = 1 \oplus x_3 \oplus x_3 x_4 \oplus x_2 x_3 \oplus \bar{x}_1 \oplus \bar{x}_1 \bar{x}_3 \oplus x_1 x_2 x_3 \oplus x_1 \bar{x}_2 \bar{x}_3 \bar{x}_4$$
$$= T_0 \oplus T_1 \oplus T_2 \oplus \cdots \oplus T_{15}$$

Where

$T_0 = 1$

$T_1 = 0 \cdot (x_4 \oplus -)$

$T_2 = 1 \cdot (x_3 \oplus 0)$

$T_3 = 1 \cdot (x_3 \oplus 0)(x_4 \oplus 0)$

$T_4 = 0 \cdot (x_2 \oplus -)$

$T_5 = 0 \cdot (x_2 \oplus -)(x_4 \oplus -)$

$T_6 = 1 \cdot (x_2 \oplus 0)(x_3 \oplus 0)$

$T_7 = 0 \cdot (x_2 \oplus -)(x_3 \oplus -)(x_4 \oplus -)$

$T_8 = 1 \cdot (x_1 \oplus 1)$

$T_9 = 0 \cdot (x_1 \oplus -)(x_4 \oplus -)$

$T_{10} = 1 \cdot (x_1 \oplus 1)(x_3 \oplus 1)$

$T_{11} = 0 \cdot (x_1 \oplus -)(x_3 \oplus -)(x_4 \oplus -)$

$T_{12} = 0 \cdot (x_1 \oplus -)(x_2 \oplus -)$

$T_{13} = 0 \cdot (x_1 \oplus -)(x_2 \oplus -)(x_4 \oplus -)$

$T_{14} = 1 \cdot (x_1 \oplus 0)(x_2 \oplus 0)(x_3 \oplus 0)$

$T_{15} = 1 \cdot (x_1 \oplus 0)(x_2 \oplus 1)(x_3 \oplus 1)(x_4 \oplus 1)$

Thus, this function can be represented using the coefficient set and termwise polarity set as

$$f\,(1011001010100011)$$

$$\{\, \delta_{T_1}(-),\ \delta_{T_2}(0),\ \delta_{T_3}(00),\ \delta_{T_4}(-),\ \delta_{T_5}(--),\ \delta_{T_6}(00),$$
$$\delta_{T_7}(---),\ \delta_{T_8}(1),\ \delta_{T_9}(--),\ \delta_{T_{10}}(11),\ \delta_{T_{11}}(---),$$
$$\delta_{T_{12}}(--),\ \delta_{T_{13}}(---),\ \delta_{T_{14}}(000),\ \delta_{T_{15}}(0111)\ \}$$

Our goal is to calculate the coefficient set of function f when one literal in the above expansion changes its polarity.

For example, if literal $\bar{x}_4$ in term $x_1\bar{x}_2\bar{x}_3\bar{x}_4$ changes its polarity, that is, $\delta_{T_{15}}(0111)$ changes to $\delta_{T_{15}}(0110)$, let $f = f_1 \oplus f_2$ where $f_1 = 1 \oplus x_3 \oplus x_3x_4 \oplus x_2x_3 \oplus \bar{x}_1 \oplus \bar{x}_1\bar{x}_3 \oplus x_1x_2x_3$ and $f_2 = x_1\bar{x}_2\bar{x}_3\bar{x}_4$. Obviously, f_1 complies with the expected polarity and f_2 does not. In order to transform f_2 to a GRM form that complies with the expected polarity, we apply the complementary expansion to f_2

recursively and have

$$
\begin{aligned}
f_2 &= x_1\bar{x}_2\bar{x}_3\bar{x}_4 \\
&= x_1\bar{x}_2\bar{x}_3 \oplus x_1\bar{x}_2\bar{x}_3 x_4 \\
&= (x_1 \oplus x_1x_2 \oplus x_1x_3 \oplus x_1x_2x_3) \oplus x_1\bar{x}_2\bar{x}_3x_4 \\
&= ((\bar{x}_1 \oplus 1) \oplus x_1x_2 \oplus (1 \oplus \bar{x}_1 \oplus \bar{x}_3 \oplus \bar{x}_1\bar{x}_3) \oplus x_1x_2x_3) \oplus x_1\bar{x}_2\bar{x}_3x_4 \\
&= ((\bar{x}_1 \oplus 1) \oplus x_1x_2 \oplus (1 \oplus \bar{x}_1 \oplus (x_3 \oplus 1) \oplus \bar{x}_1\bar{x}_3) \oplus x_1x_2x_3) \oplus x_1\bar{x}_2\bar{x}_3x_4 \\
&= 1 \oplus x_3 \oplus \bar{x}_1\bar{x}_3 \oplus x_1x_2 \oplus x_1x_2x_3 \oplus x_1\bar{x}_2\bar{x}_3x_4
\end{aligned}
$$

The decomposition of f_2 can be demonstrated by a diagram as follows:

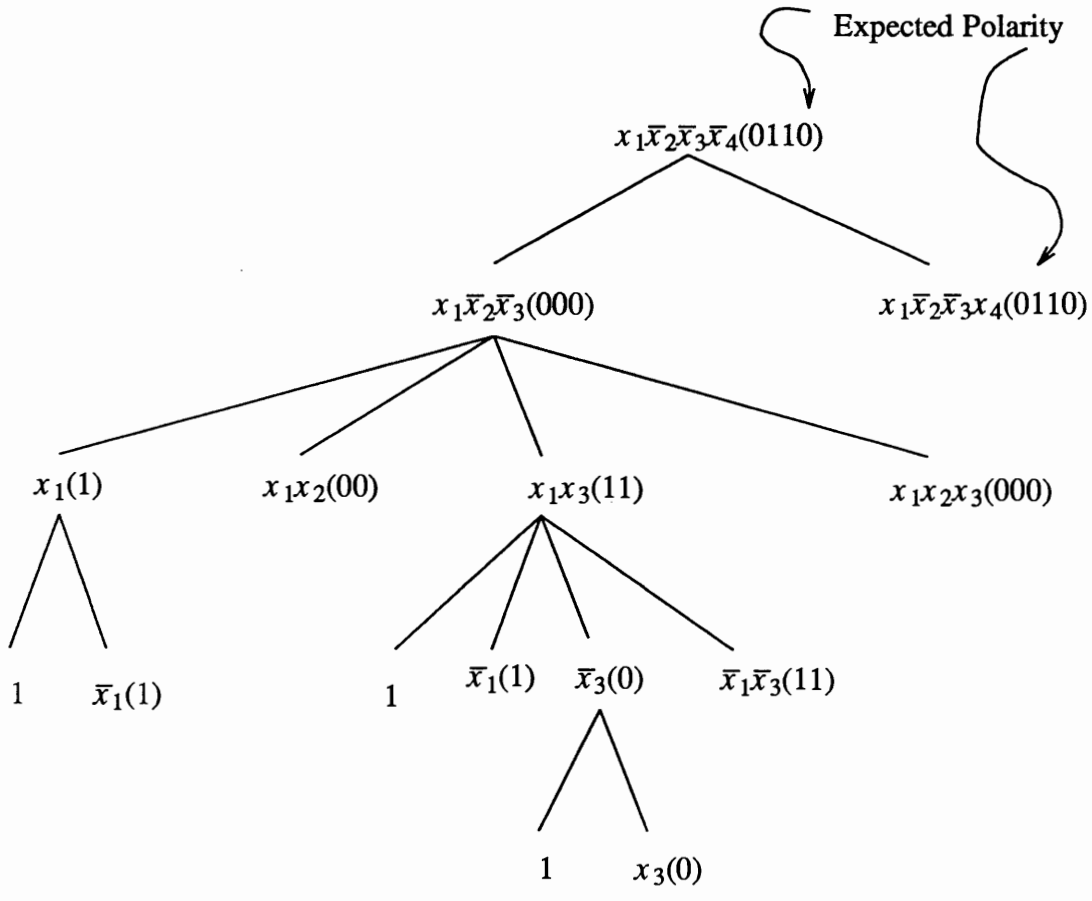


Figure 16. Termwise Complementary Expansion Diagram

Because $f_1 = 1 \oplus x_3 \oplus x_3x_4 \oplus x_2x_3 \oplus \bar{x}_1 \oplus \bar{x}_1\bar{x}_3 \oplus x_1x_2x_3$

Thus, $f = f_1 \oplus f_2 = x_3x_4 \oplus x_2x_3 \oplus \bar{x}_1 \oplus x_1x_2 \oplus x_1\bar{x}_2\bar{x}_3x_4$

The above diagram is defined as a *Termwise Complementary Expansion Diagram*. We observe that:

i. The diagram is a rooted, directed graph with nodes set V containing two types of nodes: terminal nodes and nonterminal nodes.

ii. All nodes denote terms. Each node has as reference an index. Here the reference index is the expected polarity set of the term denoted by the node.

iii. A terminal node denotes a term in which each literal complies with the reference index.

iv. A non-terminal node denotes a term which has at least one literal that does not comply with the reference index.

v. A nonterminal node has at least one descendent that is a terminal node.

vi. By exoring f_1 and all the terminal nodes in the diagram we obtain the expected adjacent polarity GRM expansion.

Algorithm 5.1.1 (Calculation of the Adjacent Polarity GRM Expansion)

Given a function in GRM expression $f = f_e = T_0 \oplus T_1 \oplus \cdots \oplus T_j \oplus \cdots \oplus T_{2^n-1}$. Where $T_j = b_j x_1^{e_1} x_2^{e_2} \cdots x_i^{e_i} \cdots x_n^{e_n}$. When the polarity of $\dot{x}_i$ in T_j is inverted, and the polarities of other literals remain unchanged. Calculate the GRM expansion f'_e from f_e under the new polarity.

1. Let $f_{2e} = T_j$, $f_{1e} = f_e \oplus f_{2e}$.

2. If $b_j = 0, f_{2e} = 0, f'_e = f_e$. Exit.
3. If $b_j \neq 0$, construct the termwise complementary expansion diagram for f_{2e}. Let $f_{2e} = T_j$ as the root. Let node i represents the descendent of T_j.
4. Compare node i with the corresponding term in f_{1e}. If two terms are identical, i is a terminal node. Check other nodes.
5. If two terms are not identical, with literals $\dot{x}_{k_1}, \dot{x}_{k_2}, \cdots$, and $\dot{x}_{k_m}$ differing in their polarities, calculate the complementary expansion of i versus $\dot{x}_{k_1}, \dot{x}_{k_2}, \cdots, \dot{x}_{k_m}$.
6. Expand the diagram until no further expansion is necessary(all descendents are terminal nodes).
7. Exoring all terminal nodes results in the GRM form of f_{2e} under the expected polarity.
8. $f'_e = f_{1e} \oplus f_{2e}$ is the GRM expansion of f under the expected polarity.

Using k number of termwise complementary expansion diagrams, one can calculate any GRM form f'_e with expected polarity from any initial GRM form f_e. Here k is the number of terms in f_e that do not comply with the expected polarity.

Example 5.4 Given is a 3-variable function in GRM form

$$f = f_e = 1 \oplus x_3 \oplus x_2 x_3 \oplus x_1 \bar{x}_2 \oplus x_1 \bar{x}_2 x_3$$

and expected polarity as shown in the following:

$$f = f'_e = b_0 \oplus b_1 \bar{x}_3 \oplus b_2 x_2 \oplus b_3 x_2 x_3 \oplus b_4 x_1 \oplus b_5 x_1 x_3 \oplus b_6 \bar{x}_1 x_2 \oplus b_7 \bar{x}_1 \bar{x}_2 \bar{x}_3$$

To calculate the coefficients of f'_e, let $f_{1e} = 1 \oplus x_2 x_3$, $f_{2e} = x_3$, $f_{3e} = x_1 \bar{x}_2$ and $f_{4e} = x_1 \bar{x}_2 x_3$.

As we can see, f_{1e} complies with the expected polarity but f_{2e}, f_{3e} and f_{4e} do not. Construct the termwise complementary expansion diagrams and calculate the GRM forms for f_{2e}, f_{3e} and f_{4e} separately as the following:

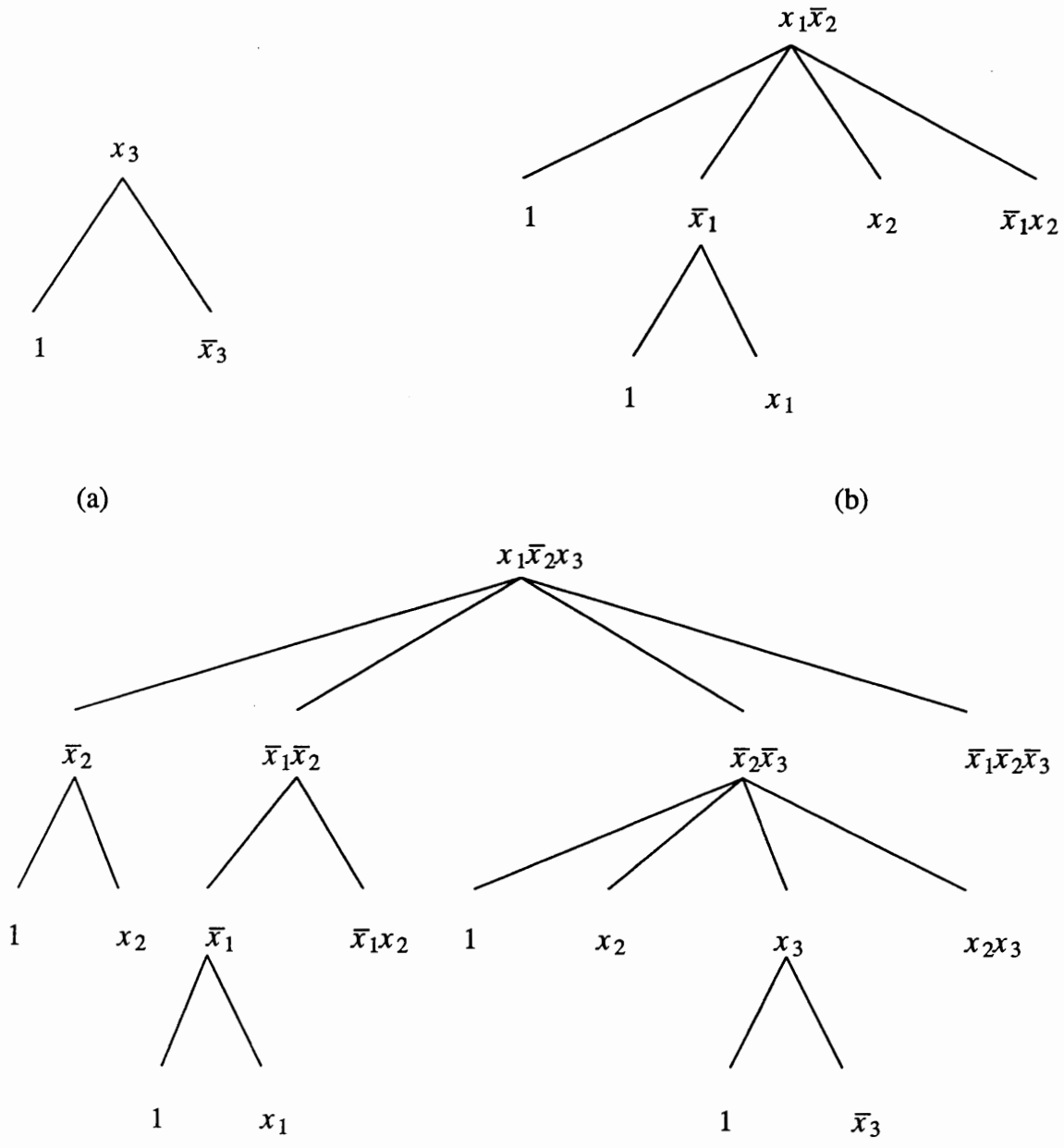


Figure 17. Termwise Complementary Expansion Diagrams for Example 5.4

From the above, we have $f_2 = 1 \oplus \bar{x}_3$, $f_3 = x_2 \oplus x_1 \oplus \bar{x}_1 x_2$, and $f_4 = \bar{x}_3 \oplus x_2 x_3 \oplus x_1 \oplus \bar{x}_1 x_2 \oplus \bar{x}_1 \bar{x}_2 \bar{x}_3$

Since $f_1 = 1 \oplus x_2 x_3$, thus,

$$\begin{aligned} f_T &= f_1 \oplus f_2 \oplus f_3 \oplus f_4 \\ &= x_2 \oplus \bar{x}_1 \bar{x}_2 \bar{x}_3 \end{aligned}$$

V.2 Exact Solution for the Minimization of GRM

By arranging the GRM polarity of function f in Gray code order and calculating the adjacent polarity expansion one by one, the minimum GRM form will be obtained by this exhaustive search. Figure 5.2 shows the data flow of the coefficient set of a 3-variable function from polarities $\delta(\dot{x}_1 \dot{x}_2 \dot{x}_3) = (000010010000)$ to $\delta(\dot{x}_1 \dot{x}_2 \dot{x}_3) = (001100010000)$.

Obviously this algorithm is very time consuming and is not practical since it has to go through all GRM expansions of a function. For a 4-variable function, it has to calculate $2^n 2^{2^{n-1}} = 2^{32}$ expansions. For a 5-variable function, it has to calculate 2^{80} expansions. If it takes 0.00001s to calculate the minimum form of a 4-variable function, then for a 5-variable one, it may take hundreds of hours.

In the next section we will present a quasi-minimum recursive algorithm which can solve much larger problems and leads to very good results.

V.3 Quasi-minimum Algorithm

This algorithm starts from an initial GRM expansion. This can be a minimized FPRM form or a GRMPRM form or any other GRM expression form. Polarity of a single literal is inverted if this inversion reduces the number of terms. This procedure is carried out iteratively until no further improvement can be achieved.

Given a n-variable function in GRM form:

$$f = f_e = T_0 \oplus T_1 \oplus \cdots \oplus T_j \oplus \cdots \oplus T_{2^n-1}$$

Where $T_j = b_j \dot{x}_1^{e_1} \dot{x}_2^{e_2} \cdots \dot{x}_i^{e_i} \cdots \dot{x}_n^{e_n}$

The coefficient set of the above GRM form is

$$f(b_0 b_1 \cdots b_j \cdots b_{2^n-1})$$

And the termwise polarity set of the above GRM form is

$$\{\delta_{T_0}, \delta_{T_1}, \cdots \delta_{T_j}, \cdots \delta_{T_{2^n-1}}\}$$

The cost of the above expression is:

$$cost(f_i) = \sum_{j=0}^{2^n-1} b_j$$

If we change the polarity set $\delta_{T_{2^n-1}}$ in Gray code order and keep the other termwise polarity sets $\delta_{T_0}, \delta_{T_1}, \cdots \delta_{T_j}, \cdots, \delta_{T_{2^n-2}}$ unchanged, by calculating all the adjacent polarity expansions, we have an optimal expansion $f = f'_e$ with the best cost function $cost(f'_e) \le cost(f_e)$ under this polarity arrangement. The termwise polarity set of f'_e

is $\{\delta_{T_0},\delta_{T_1},\cdots\delta_{T_j},\cdots\delta'_{T_{2^n-1}}\}$. If we use f'_e and $\{\delta_{T_0},\delta_{T_1},\cdots\delta_{T_j},\cdots\delta'_{T_{2^n-1}}\}$ as the input vector and apply the above procedure to the polarity set $\delta_{T_{2^n-2}}$, again we have another optimal expansion $f = f''_e$ with the cost function $cost(f''_e) \le cost(f'_e)$. Apply the above procedure to all termwise polarity sets, finally we obtain the optimal GRM expression of this pass $f_{pass\,1}$ with the polarity set $\{\delta'_{T_0},\delta'_{T_1},\cdots\delta'_{T_j},\cdots\delta'_{T_{2^n-1}}\}$. If the cost of $f_{pass\,1}$ is better than the cost of the initial expression, then the above procedure is carried out again. This is done recursively until no further improvement can be achieved. The algorithm is presented as follows:

Algorithm 5.1: (quasi-minimization of the GRM expansion)

1. f_e be the initial GRM form. Let $j = 2^n - 1$. $f_{\min} = f_e$.
2. If $b_j = 0$, no matter how we select the polarities of literals in T_j, the cost function remains unchanged. $j = j - 1$. Goto step 2.
3. If $b_j \neq 0$. Arrange the termwise polarity set δ_{T_j} in Gray code order, calculate all GRM forms under each polarity set of term T_j. Select the one which has the best cost function. Let f'_e be the resulting optimal GRM form. Then the termwise polarity set of f'_e is

   $$\{\delta_{T_0},\delta_{T_1},\cdots\delta'_{T_j},\cdots\delta'_{T_{2^n-1}}\}$$

   Note that f'_e may be the same as f_e if the original polarity set of term T_j is the optimal selection.
4. Let $f_e = f'_e$, $j = j - 1$. $f_{\min} = f'_e$. Goto to step 2.
5. After applying the above procedure to all terms we have a GRM form $f_{pass\,1}$ with the optimal cost of this pass.
6. If $cost(f_{pass\,1}) < cost(f_{\min})$, let $f_{\min} = f_{pass\,1}$, $f_e = f_{pass\,1}$, goto step 1.

7. If $cost(f_{pass\,1}) = cost(f_e)$, exit.

From the above, we can see that Algorithm 5.1 calculates and searches all GRM expansions within the polarity set of each term, thus at least local exact minimum solutions can be obtained at terms level. Since it is a heuristic algorithm, the general effectiveness must be verified by benchmark results, which are shown in chapter VI. Figure 18 shows the procedure of Algorithm 5.1.

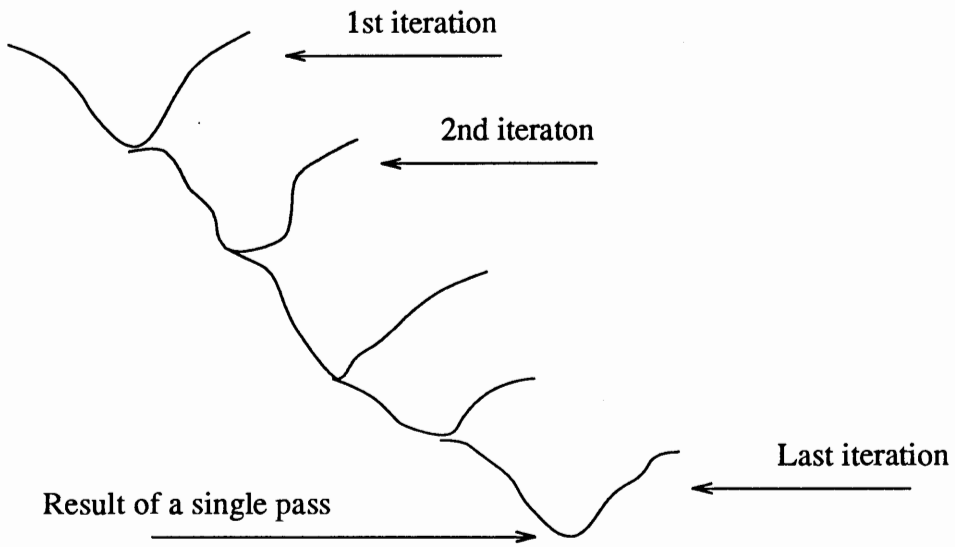


Figure 18. Illustration of Algorithm 5.1

We also observe that when the coefficient of a term is zero, the cost functions of all the GRM expansions under each polarity set of this term will be the same. Thus, the calculation of these expansions is not necessary. Since after each iteration, the number of terms in the optimal expansion is reduced, and this optimal expansion is used as the input function to the next iteration. Thus, the computation time is decreased.

As an example, the exact minimum GRM expansion for $f = 1 \oplus x_3 \oplus \bar{x}_1\bar{x}_3 \oplus x_1x_2 \oplus x_1x_2x_3 \oplus x_1\bar{x}_2\bar{x}_3x_4$ is $f_e = x_1\bar{x}_2\bar{x}_3\bar{x}_4$. In order to reach this solution, the exact algorithm has to calculate $2^n 2^{n-1} = 2^{32}$ GRM expansions. For the quasi-minimum one, only $2^n = 16$ expansions are calculated before the exact solution is found. Experimental results show that the computation time of the quasi-minimum algorithm depends mainly on the complexity, that is, the number of input variables and the number of terms in the input function.

For an n-variable function, the maximum number of literals in a product term is n. Thus for each iteration at most 2^n GRM expansions are calculated. For an input function with t product terms, the upperbound of expansions calculated is $t \cdot 2^n$. Therefor, the time complexity of Algorithm 5.1 is $O(t \cdot 2^n)$. The space complexity is $O(n \cdot 2^{n+1})$.

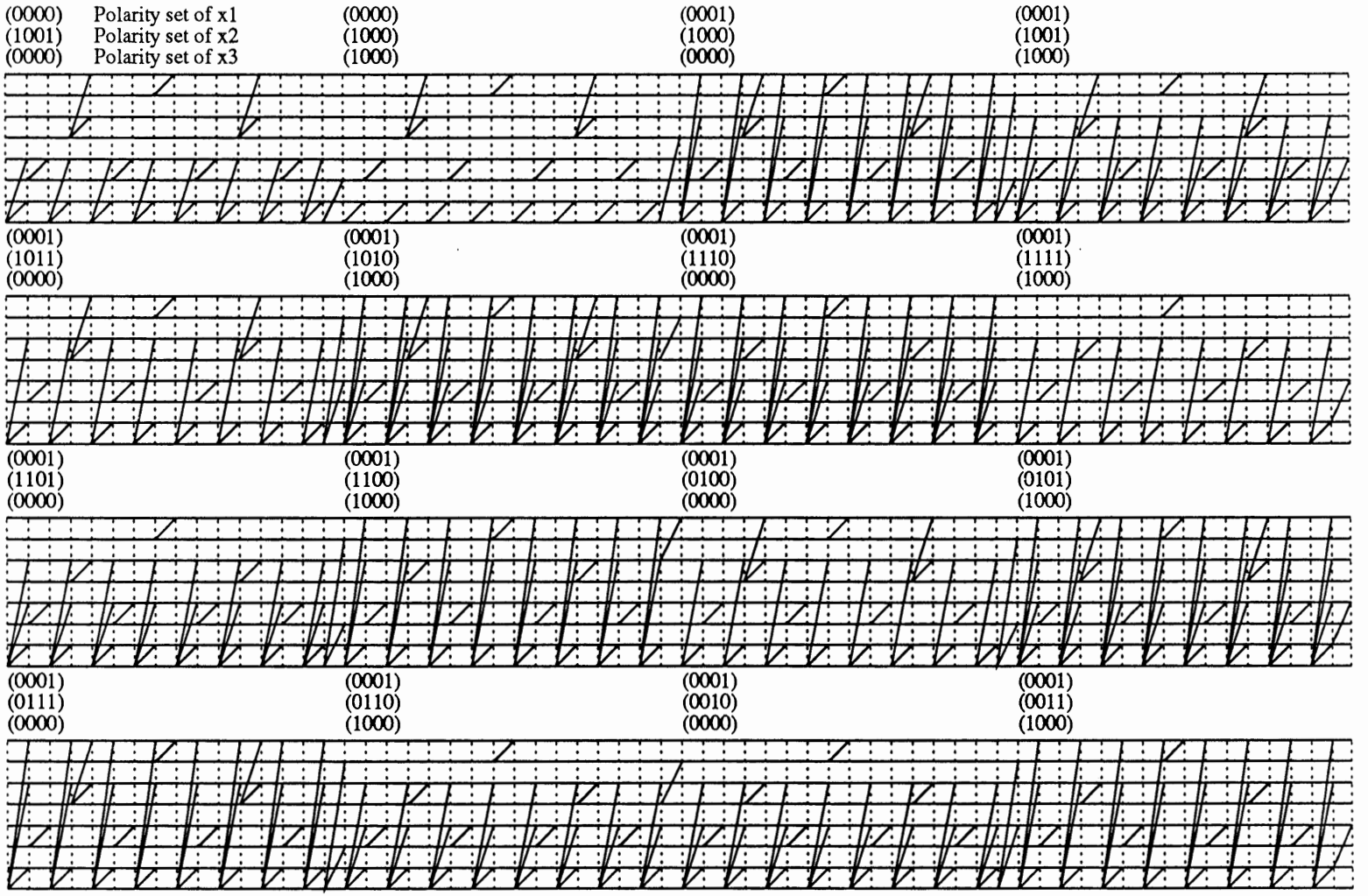


Figure 19. Calculation of GRM Expansions for a 3-variable Function (from Polarity (0000 1001 0000) to (0001 0011 1000))

CHAPTER VI

PROGRAMS AND RESULTS

The algorithms 4.2 and 5.2 presented in the previous sections have been implemented in programs named exPMPRM, and qGRM. They read in the input file and output the exact minimum PMPRM and quasi-minimum GRM, respectively. Both the input and output files are in the Espresso format. The cyclic code is used through out the programs for the polarity encoding. Over 100 sample functions from MCNC benchmark have been tested and the results are very encouraging.

VI.1 Espresso Format

The Espresso format is a two-level description of a Boolean function. It is a character matrix with keywords embedded in the input to specify the size of the matrix and the logical format of the input function. A logic function given in Espresso format is as follows:

```
.i  18
.o  5
.p  7

10-01-010101-1-010  10-10
1-11-111-1-1100000  01-01
01-1001-01010101-1  01-0-
..                  ..
..                  ..
-01010-0101-1010-1  1-110

.end
```

In the above file:

.i 18 Specifies the number of input variables(18).

.o 5 Specifies the number of output functions(5).

.p 7 Specifies the number of product terms(7).

.end Marks the end of the input logic.

The matrix in the above file represents the input and output functions. A row in the input matrix(left one) represents a product term in the input function. The literals of a variable in the input function are represented by the corresponding column in the input matrix. A "0" is a positive literal and a "1" is a negative literal. A "-" denotes a nonexisting literal of a variable in the term represented by the corresponding row. A column in the output matrix represents the coefficient set of the output function. Each character in the output matrix represents the coefficient of the corresponding term in the input array. Since we address our problem only for the single output functions, only one column exists in the output matrix and only those terms with cofficients equal to 1 are shown in the file.

VI.2 Gray Code versus Polarity Encoding

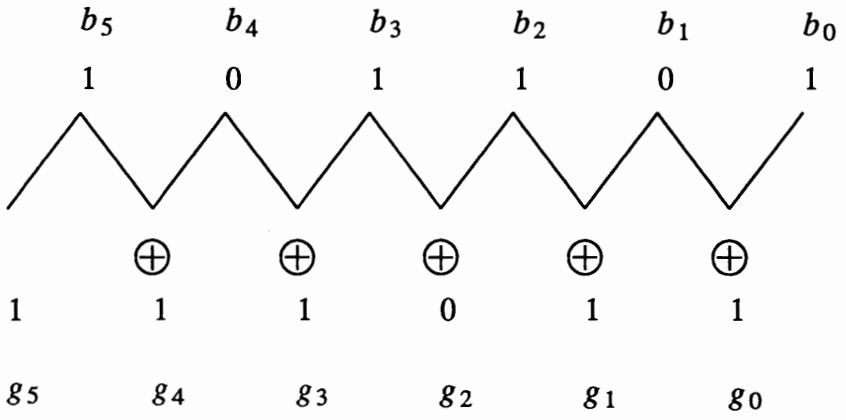
As we stated in the previous chapter, the encoding of the polarity set has to use codes in which all successive code words differ in only one digit. Codes that have such a property are refered to as *cyclic* codes. A particularly important cyclic code is the *Graycode*. The feature that makes this cyclic code useful is the simplicity of the procedure for converting from the binary number system into the Gray code.

Let $g_n \cdots g_2 g_1 g_0$ denote a code word in the (n+1)st-bit Gray code, and let $b_n \cdots b_2 b_1 b_0$ designate the corresponding binary number, where the subscripts 0 and n denote the least significant and the most significant digits, respectively. Then the *ith* digit g_i can be obtained from the corresponding binary number as follows:

$$g_i = b_i \oplus b_{i+1} \quad 0 \le i \le n-1$$

$$g_n = b_n$$

For example, the Gray code word which corresponds to the binary number 101101 is found to be 111011 in the following manner:



If we use the binary code as the counting vector, then during every increment, only one bit of this binary vector would change from "0" to "1". Inverting the corresponding bit in the polarity vector of the target function results in a cyclic code that meets the need of our algorithm. If the input polarity vector is a positive polarity(zero polarity), then the resulting polarity encoding is exactly the Gray code.

Table 2 shows a four bit polarity encoding in our algorithms. The second and the first columns are the binary counting vector and the corresponding decimal number. The third column is the corresponding Gray code. This can be used as polarity encoding

if the algorithm begins from a PPRM. The fourth column is the polarity encoding that starts from an arbitary polarity set(Here the input polarity vector is (0101).

Table 2. Polarity Encoding

1	2	3	4	5
0	0000	0000	0101	5
1	0001	0001	0100	4
2	0010	0011	0110	6
3	0011	0010	0111	7
4	0100	0110	0011	3
5	0101	0111	0010	2
6	0110	0101	0000	0
7	0111	0100	0001	1
8	1000	1100	1001	9
9	1001	1101	1000	8
10	1010	1111	1010	10
11	1011	1110	1011	11
12	1100	1010	1111	15
13	1101	1011	1110	14
14	1110	1001	1100	12
15	1111	1000	1101	13

VI.3 Pseudo Code for Programs exPMPRM and qGRM

In our program, Counting is implemented in function $k = count()$ where k is the bit that changes from "0" to "1" in this increment. The pseudo-code of $k = count()$ is shown as the following:

```
/**** counting function *******/

count()
{
   for(i=1;i<=bit_num;i++){
        /* bit_num is the number of bit in the polarity set. In exPMPRM,
           bit_num=n. In quGRM, bit_num is the number of variables in term T_j */
      if(counter[i]==0){
         counter[1]=1;
         return(i);
      }
      else
         counter[i]=0;
   }
}
```

Following is the pseudo code of exPMPRM

```
/**** exact minimum PMPRM *******/

exPMPRM()
{
   read in input file f_i in FPRM expression;
   f_min = f_i;

   for(i=1;i<=n;i++){          /* go through all variables */
      f_e = f_i;
      fp = polarity of f_e;  /* fp is a n-bit polarity vector of FPRM. the polarity of x_i
                                is arranged at the 1st bit, the least sigfinicant bit */
      do{
         k=count();
         inverse the k^th -bit of fp;

         for(j=0;j<=2^n-1;j++){  /* calculate the adjacent polarity expansion
                                    f'_e from f_e */
            if(T_{j+2^{n-i}} contains x_i)
               b'_j = b_j ⊕ b_{j+2^{n-i}};
            else
               b'_j = b_j;
         }
         if(cost(f'_e) < cost(f_min))
            f_min = f'_e
         if(k==1){
            for(j=0;j<=2^n-1;j++){   /* calculate the optimal PMPRM
                                        f''_e from f'_e and f_e */
               if(b_j = 1 and b'_j = 0)
                  b''_j = b'_j;
               else
```

```
                b''_j = b_j;
            }
            if(cost(f''_e) < cost(f_min))
                f_min = f''_e
        }
        f_e = f'_e;
    }while(k!=n+1);
}
output f_min;
}
```

Before we present the pseudo code of the qGRM, let us first explain how to obtain the complementary expansion of term T. From section 5.2, we know that the complementary expansion of T is a FPRM with all the coefficients equal to "1" and the polarities of variables be the complement of that in T. By constituting the variables in T, we have all the subsets of these variables and each of these subsets represents a term in the complementary expansion. For example, if $T = \dot{a}\dot{b}\dot{c}$, then the set of constitution is $(1,(\dot{a}),(\dot{b}),(\dot{c}),(\dot{a},\dot{b}),(\dot{a},\dot{c}),(\dot{b},\dot{c}),(\dot{a},\dot{b},\dot{c}))$. In the program the constitution is done by function *consttn*() which accepts a set of decision variables and returns the constitution results, the descendent nodes.

We use two stacks in the construction of the termwise complementary expansion diagram and computation of the adjacent polarity GRM expansion. Stack A is used as a dynamic memory to storage the generated descendent nodes, and all the terminal nodes are put in stack B. The precedure is described as follows:

1. Push the root into stack A. This is the term in which the polarity of literal x_i changed.
2. Pop stack A, we have node i. Check i. If i is a terminal node, push i to stack B. If i is a nonterminal node, calculate the complementary expansion of i, expand the diagram and push the descendents of i into stack A.
3. If stack A is not empty, go to step 2.

4. If stack A is empty, pop all nodes in stack B and perform the exor operation with the corresponding terms in f.

The pseudo code of qGRM is shown as the following. The function *check*() accepts a node(polarity array of a term) and then compares with the polarity vector. If they are identical, returns a NULL. Otherwise it returns a set of decision variables that do not comply with the expect polarities.

```
/**** quasi-minimum GRM *******/

qGRM()
{
    read f_i in FPRM;
    f_min=f_i;

    while(improvement == true){

        for(j=2^n-1; j>=1;j--){   /* start a pass recursion */
            f_e =f_min;
            pv = polarity vector;
            flag=number of variables in T_j;
            do{
                k=count();
                inverse the k^th bit in pv;
                push T_j to stk_A;  /* T_j is the root */

                while(stk_A != NULL){
                    node_i = pop(stk_A);
                    decision_vars = check(node_i);
                    if(decision_vars != NULL){
                        descnnts=contttn(decision_vars);
                        push descnnts to stk_A;
                    }
                    else
                        push node_i to stk_B;   /* node_i is a terminal node */
                }
                while(stk_B != NULL)
                    f_e =f_e ⊕ pop(stk_B);
                if(cost(f_e)<cost(f_min))
                    f_min=f_e;
            }while(k!=flag+1);
        }
        output f_min;
    }
}
```

VI.4 Experimental Results

The exact minimization algorithm for PMPRM and the quasi-minimal method for GRM were tested for more than 100 MCNC benchmark functions. As most of the benchmarks are multi-output, the BLIF format of the functions was used to generate single output components of these functions for testing. These functions were first run through the minimizer CGRMIN[12], which gave the minimum solutions of FPRM. Then the minimum FPRMs were used as the input functions to our programs exGPMPRM and qGRM. The results of these programs are shown in Table 3 through Table 6.

In these tables, the column headed *function* denotes the name of the function from the benchmark. Var stands for the number of input variables. The columns headed PMPRM, qGRM, ESPRE and EXCSM, CGRM are the numbers of terms in the output files of minimizers exPMPRM, qGRM, ESPRESSO and EXORCISM[36], and CGRMIN[12], respectively.

For the functions tested, the exPMPRM and qGRM were found to be comparable to the results of other minimizers. The results of both these two programs are never worse than the CGRM, which is the exact solution for the minimization of FPRM. Depending on the functions, EXORCISM, a minimizer for ESOP circuit forms, can be a better choice for realization while for others it is the qGRM that gives the better alternative. For the 110 functions overall, qGRM had 11 results better than EXORCISM, and for 52 functions the two minimizer gave the same results. Please note here that since both the PMPRM and the GRM are sub-families of the ESOP, thus their results will not necessarily be better than that of the EXORCISM. As to the two level AND/OR minimizer ESPRESSO, for 110 functions overall, while ESPRESSO found 1336 terms, this

number for qGRM was found to be 918.

Figure 19 gives the plot of exPMPRM execution time versus the number of variables in the input function. Figure 20 gives the scatter plot of the number of input variables versus program qGRM execution time, while Figure 21 is the scatter plot of the number of terms in the input function versus qGRM execution time. Here NT is the number of terms in the input FPRM expansion. By comparing Figure 20 and Figure 21 we can see that the execution time of the quasi-minimum GRM minimization algorithm depends more on the number of input terms, than on the number of input variables.

Table 3. Benchmark results for PMPRM, ESOP, and FPRM

Function	Var	PMPRM	ESPRE	CGRMIN	Time
5x3	7	14	18	19	0.22
5x5	7	6	10	7	0.20
5xp11	7	9	7	12	0.22
9sym	9	139	86	173	3.9
con1	7	9	6	12	0.22
f52	8	14	18	19	0.85
f53	8	10	14	11	0.83
misex47	11	13	4	18	72.22
misex54	6	12	11	16	0.05
misex58	6	5	6	7	0.07
misex62	10	40	7	50	16.45
misex63	10	62	7	84	16.48
misex64	10	21	4	28	16.22
rd532	5	5	16	5	0.03
rd732	7	7	64	7	0.22
rd842	8	8	128	8	0.83
sao22	10	37	20	58	16.58
sao23	10	35	22	47	16.52
xor5	5	5	16	5	0.00
z42	7	9	28	9	0.18
z4ml	7	22	7	25	0.20
Total terms		482	499	620	

Table 4. Benchmark results for qGRM and ESPRESSO

Function	Var	qGRM	ESPRE	Function	Var	qGRM	ESPRE
5x01	7	8	7	bw27	5	5	4
5x1	7	23	65	bw3	5	4	4
5x10	7	2	3	bw4	5	5	6
5x2	7	15	11	bw5	5	5	4
5x3	7	12	18	bw6	5	3	5
5x4	7	9	14	bw7	5	5	6
5x5	7	5	10	bw8	5	3	4
5x6	7	4	5	bw9	5	4	3
5x7	7	2	3	9sym	9	129	86
5xp1	7	20	65	cm152a	11	8	8
bw01	5	7	6	con1	7	10	9
bw10	5	2	3	con11	7	5	4
bw11	5	2	2	con12	7	5	5
bw12	5	3	4	f21	4	3	3
bw13	5	2	3	f22	4	3	3
bw14	5	4	4	f23	4	3	3
bw15	5	2	3	f24	4	3	3
bw16	5	3	4	f501	8	17	23
bw17	5	2	3	f52	8	12	76
bw18	5	5	6	f53	8	9	18
bw19	5	4	5	f54	8	5	10
bw2	5	3	3	f55	8	4	5
bw20	5	5	6	f56	8	2	5
bw21	5	3	5	f57	8	2	2
bw23	5	5	6	majority	5	5	5
bw24	5	6	6	misex20	6	19	7
bw25	5	4	5	misex21	6	9	11
bw26	5	6	5	misex22	6	6	6

Table 4.(cont.) Benchmark results for qGRM and ESPRESSO

Function	Var	qGRM	ESPRE	Function	Var	qGRM	ESPRE
misex23	6	5	6	misex69	4	2	3
misex24	4	2	3	misex70	5	6	6
misex25	6	6	6	misex71	5	5	5
misex26	6	5	6	misex72	4	2	3
misex27	5	6	6	misex73	4	2	3
misex41	5	6	6	misex74	4	2	3
misex42	4	2	3	misex75	5	6	6
misex43	4	2	3	rd53	5	12	31
misex44	4	2	3	rd531	5	5	5
misex45	5	5	5	rd532	5	5	16
misex47	11	7	4	rd533	5	10	10
misex48	6	8	11	rd731	7	21	42
misex49	6	7	6	rd732	7	7	64
misex50	6	5	6	rd733	7	35	35
misex51	4	2	3	rd842	8	8	128
misex52	6	6	6	rd844	8	70	70
misex53	6	4	6	sam	5	7	6
misex54	6	9	11	sao21	10	13	10
misex55	6	6	6	sao22	10	19	20
misex56	6	5	6	sao23	10	16	23
misex57	6	6	6	sao24	10	24	21
misex58	6	5	6	xor5	5	5	16
misex62	10	17	7	z41	7	15	15
misex63	10	20	7	z42	7	9	28
misex64	10	6	4	z43	7	5	12
misex66	5	5	5	z44	7	3	4
misex67	4	2	3	Total term numbers		918	1336
misex68	4	2	3				

Table 5. Benchmark results for qGRM and CGRMIN

Function	Var	qGRM	CGRM	Function	Var	qGRM	CGRM
5x01	7	8	12	bw27	5	5	6
5x1	7	23	61	bw3	5	4	6
5x10	7	2	2	bw4	5	5	7
5x2	7	15	30	bw5	5	5	8
5x3	7	12	19	bw6	5	3	6
5x4	7	9	11	bw7	5	5	10
5x5	7	5	7	bw8	5	3	3
5x6	7	4	4	bw9	5	4	5
5x7	7	2	2	9sym	9	129	173
5xp1	7	20	61	cm152a	11	8	27
bw01	5	7	12	con1	7	10	17
bw10	5	2	4	con11	7	5	9
bw11	5	2	2	con12	7	5	8
bw12	5	3	6	f21	4	3	3
bw13	5	2	4	f22	4	3	3
bw14	5	4	8	f23	4	3	3
bw15	5	2	6	f24	4	3	3
bw16	5	3	4	f501	8	17	31
bw17	5	2	4	f52	8	12	19
bw18	5	5	5	f53	8	9	11
bw19	5	4	7	f54	8	5	7
bw2	5	3	4	f55	8	4	4
bw20	5	5	7	f56	8	2	2
bw21	5	3	5	f57	8	2	2
bw23	5	5	9	majority	5	5	7
bw24	5	6	7	misex20	6	19	62
bw25	5	4	8	misex21	6	9	16
bw26	5	6	8	misex22	6	6	10

Table 5.(cont.) Benchmark results for qGRM and CGRMIN

Function	Var	qGRM	CGRM	Function	Var	qGRM	CGRM
misex23	6	5	7	misex69	4	2	2
misex24	4	2	2	misex70	5	6	9
misex25	6	6	9	misex71	5	5	6
misex26	6	5	7	misex72	4	2	2
misex27	5	6	9	misex73	4	2	2
misex41	5	6	9	misex74	4	2	2
misex42	4	2	2	misex75	5	6	9
misex43	4	2	2	rd53	5	12	20
misex44	4	2	2	rd531	5	5	5
misex45	5	5	6	rd532	5	5	5
misex47	11	7	18	rd533	5	10	10
misex48	6	8	16	rd731	7	21	2
misex49	6	7	10	rd732	7	7	7
misex50	6	5	7	rd733	7	35	35
misex51	4	2	2	rd842	8	8	8
misex52	6	6	9	rd844	8	70	70
misex53	6	4	6	sam	5	7	12
misex54	6	9	16	sao21	10	13	36
misex55	6	6	10	sao22	10	19	52
misex56	6	5	7	sao23	10	16	47
misex57	6	6	9	sao24	10	24	55
misex58	6	5	7	xor5	5	5	5
misex62	10	17	50	z41	7	15	15
misex63	10	20	84	z42	7	9	9
misex64	10	6	28	z43	7	5	5
misex66	5	5	6	z44	7	3	3
misex67	4	2	2	Total term numbers		918	1544
misex68	4	2	2				

Table 6. Benchmark results for qGRM and EXORCISM

Function	Var	qGRM	EXCSM	Function	Var	qGRM	EXCSM
5x01	7	8	6	bw27	5	5	4
5x1	7	23	33	bw3	5	4	4
5x10	7	2	3	bw4	5	5	4
5x2	7	15	10	bw5	5	5	4
5x3	7	12	9	bw6	5	3	4
5x4	7	9	7	bw7	5	5	5
5x5	7	5	5	bw8	5	3	3
5x6	7	4	3	bw9	5	4	3
5x7	7	2	2	9sym	9	129	58
5xp1	7	20	32	cm152a	11	8	8
bw01	5	7	5	con1	7	10	9
bw10	5	2	3	con11	7	5	5
bw11	5	2	2	con12	7	5	4
bw12	5	3	4	f21	4	3	3
bw13	5	2	3	f22	4	3	3
bw14	5	4	4	f23	4	3	3
bw15	5	2	3	f24	4	3	3
bw16	5	3	3	f501	8	17	11
bw17	5	2	3	f52	8	12	9
bw18	5	5	5	f53	8	9	7
bw19	5	4	4	f54	8	5	5
bw2	5	3	3	f55	8	4	3
bw20	5	5	5	f56	8	2	2
bw21	5	3	3	f57	8	2	2
bw23	5	5	5	majority	5	5	5
bw24	5	6	5	misex20	6	19	7
bw25	5	4	4	misex21	6	9	8
bw26	5	6	5	misex22	6	6	6

Table 6.(cont.) Benchmark results for qGRM and EXORCISM

Function	Var	qGRM	EXCSM	Function	Var	qGRM	EXCSM
misex23	6	5	5	misex69	4	2	2
misex24	4	2	2	misex70	5	6	5
misex25	6	6	5	misex71	5	5	4
misex26	6	5	5	misex72	4	2	2
misex27	5	6	5	misex73	4	2	2
misex41	5	6	5	misex74	4	2	2
misex42	4	2	2	misex75	5	6	5
misex43	4	2	2	rd53	5	12	15
misex44	4	2	2	rd531	5	5	5
misex45	5	5	4	rd532	5	5	5
misex47	11	7	4	rd533	5	10	8
misex48	6	8	8	rd731	7	21	15
misex49	6	7	6	rd732	7	7	7
misex50	6	5	5	rd733	7	35	24
misex51	4	2	2	rd842	8	8	8
misex52	6	6	5	rd844	8	70	32
misex53	6	4	5	sam	5	7	5
misex54	6	9	8	sao21	10	13	10
misex55	6	6	6	sao22	10	19	13
misex56	6	5	5	sao23	10	16	12
misex57	6	6	5	sao24	10	24	11
misex58	6	5	5	xor5	5	5	5
misex62	10	17	7	z41	7	15	15
misex63	10	20	7	z42	7	9	9
misex64	10	6	4	z43	7	5	5
misex66	5	5	4	z44	7	3	3
misex67	4	2	2	Total term numbers		918	725
misex68	4	2	2				

Table 7. Computation time of qGRM

Function	Var	qGRM	Time	Function	Var	qGRM	Time
5x01	7	8	0.63	bw27	5	5	0.07
5x1	7	23	12.45	bw3	5	4	0.05
5x10	7	2	0.05	bw4	5	5	0.07
5x2	7	15	10.2	bw5	5	5	0.07
5x3	7	12	1.12	bw6	5	3	0.02
5x4	7	9	0.42	bw7	5	5	0.05
5x5	7	5	0.15	bw8	5	3	0.03
5x6	7	4	0.10	bw9	5	4	0.03
5x7	7	2	0.07	9sym	9	129	15.3
5xp1	7	20	12.67	cm152a	11	8	4.93
bw01	5	7	0.10	con1	7	10	0.32
bw10	5	2	0.02	con11	7	5	0.27
bw11	5	2	0.03	con12	7	5	0.13
bw12	5	3	0.03	f21	4	3	0.01
bw13	5	2	0.02	f22	4	3	0.02
bw14	5	4	0.05	f23	4	3	0.02
bw15	5	2	0.03	f24	4	3	0.02
bw16	5	3	0.03	f501	8	17	12.63
bw17	5	2	0.02	f52	8	12	1.43
bw18	5	5	0.05	f53	8	9	0.62
bw19	5	4	0.08	f54	8	5	0.30
bw2	5	3	0.05	f55	8	4	0.22
bw20	5	5	0.06	f56	8	2	0.12
bw21	5	3	0.02	f57	8	2	0.12
bw23	5	5	0.08	majority	5	5	0.13
bw24	5	6	0.05	misex20	6	19	77.43
bw25	5	4	0.12	misex21	6	9	0.22
bw26	5	6	0.07	misex22	6	6	0.18

Table 7(cont.). Computation time of qGRM

Function	Var	qGRM	Time	Function	Var	qGRM	Time
misex23	6	5	0.22	misex69	4	2	0.02
misex24	4	2	0.02	misex70	5	6	0.07
misex25	6	6	0.23	misex71	5	5	0.07
misex26	6	5	0.22	misex72	4	2	0.02
misex27	5	6	0.08	misex73	4	2	0.02
misex41	5	6	0.07	misex74	4	2	0.02
misex42	4	2	0.00	misex75	5	6	0.07
misex43	4	2	0.02	rd53	5	12	0.35
misex44	4	2	0.00	rd531	5	5	0.08
misex45	5	5	0.07	rd532	5	5	0.00
misex47	11	7	116.22	rd533	5	10	0.03
misex48	6	8	0.33	rd731	7	21	0.42
misex49	6	7	0.25	rd732	7	7	0.17
misex50	6	5	0.23	rd733	7	35	1.07
misex51	4	2	0.02	rd842	8	8	0.37
misex52	6	6	0.27	rd844	8	70	4.05
misex53	6	4	0.15	sam	5	7	0.08
misex54	6	9	0.18	sao21	10	13	40.78
misex55	6	6	0.22	sao22	10	19	70.43
misex56	6	5	0.17	sao23	10	16	133.67
misex57	6	6	0.22	sao24	10	24	77.08
misex58	6	5	0.18	xor5	5	5	0.03
misex62	10	17	63.37	z41	7	15	0.43
misex63	10	20	78.12	z42	7	9	0.22
misex64	10	6	28.25	z43	7	5	0.13
misex66	5	5	0.07	z44	7	3	0.07
misex67	4	2	0.02				
misex68	4	2	0.03				

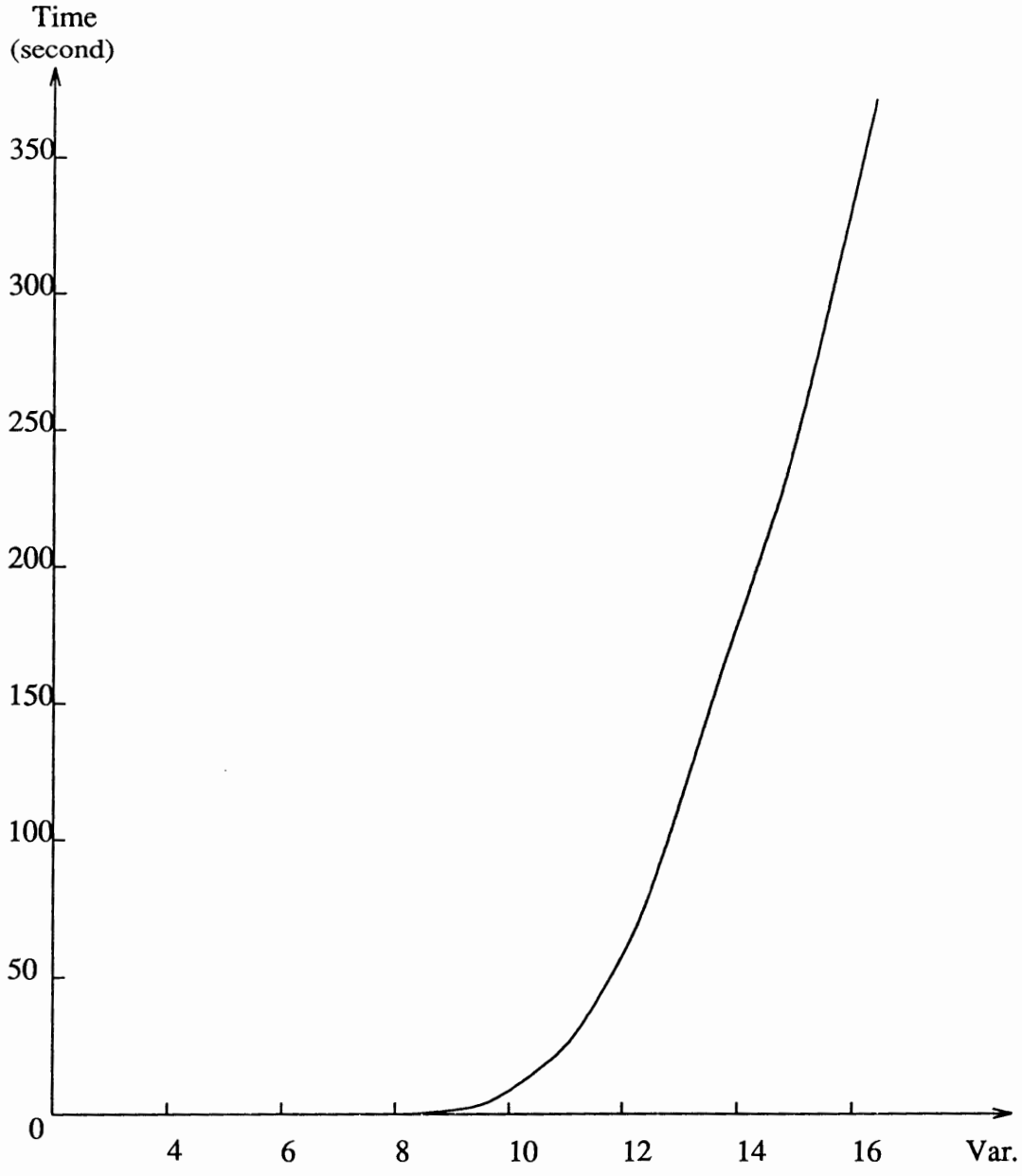


Figure 20. Plot of Number of Input Variables versus exPMPRM Execution Time

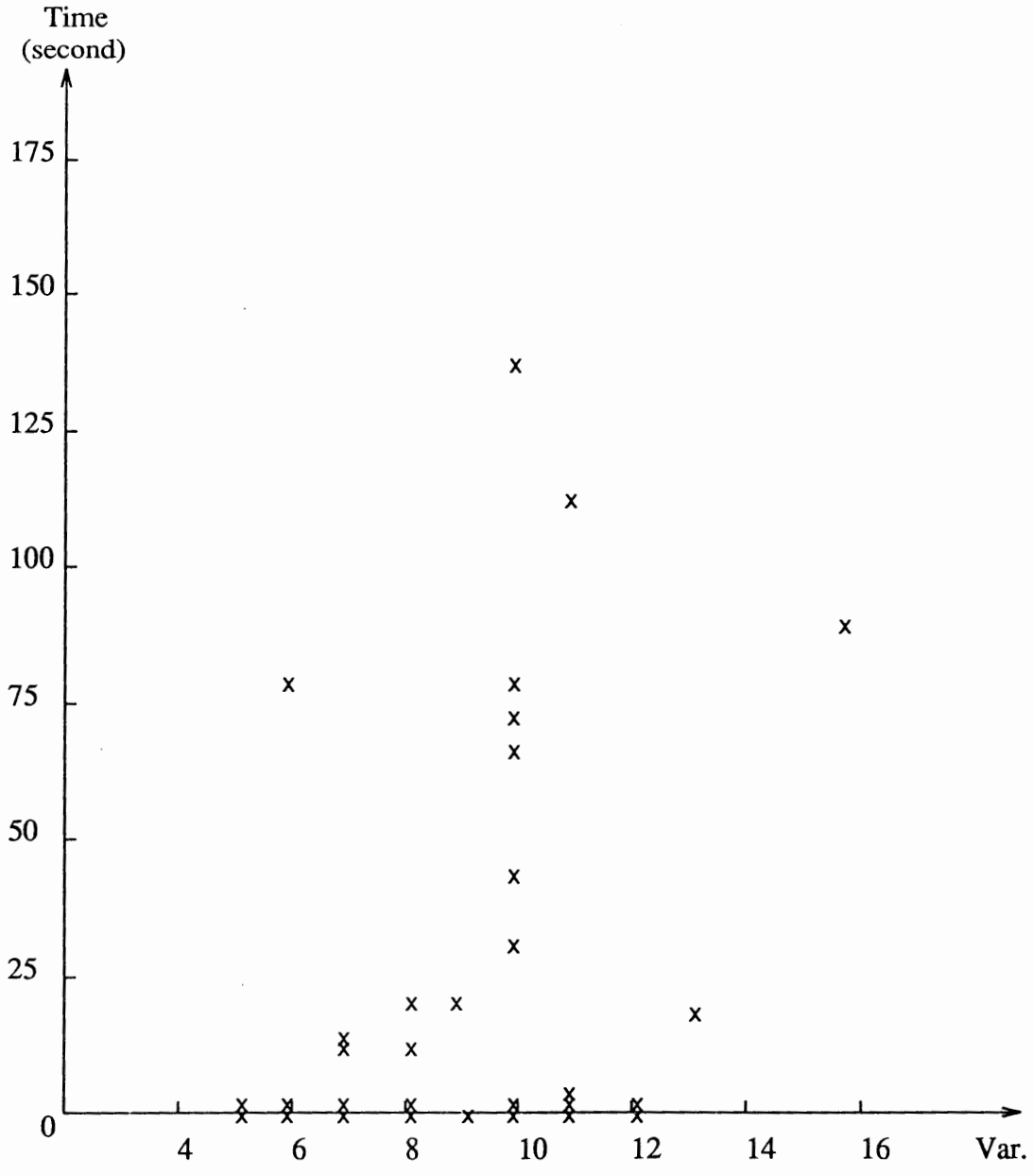


Figure 21. Scatter Plot of Number of Variables versus qGRM Execution Time

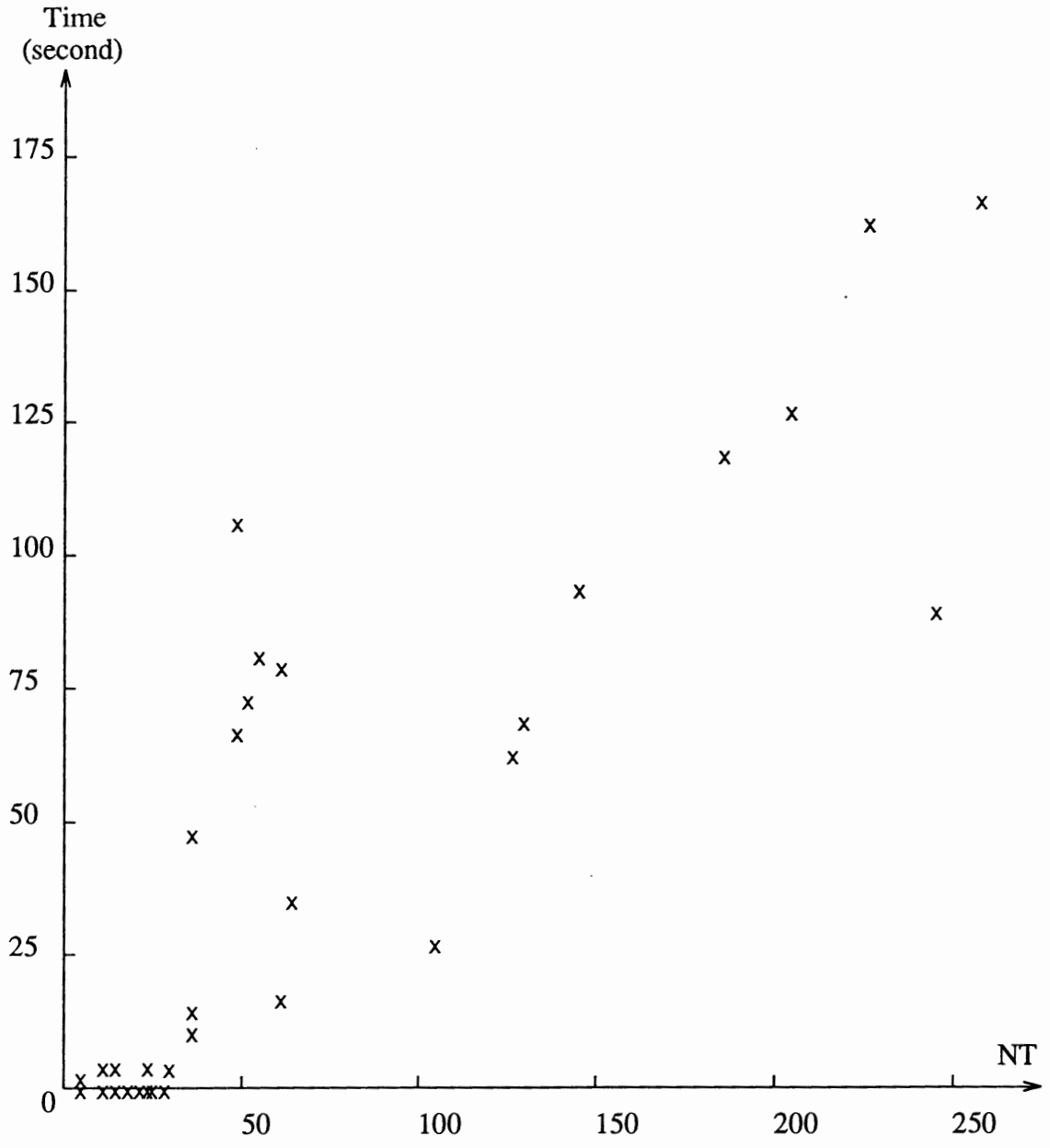


Figure 22. Scatter Plot of Number Input Terms versus qGRM Execution Time

CHAPTER VII

CONCLUSION AND FUTURE WORK

VII.1 Conclusion

Several classes of AND-EXOR circuit expressions have been defined and their relationships have been shown. A new class of AND-EXOR circuits, the Partially Mixed Polarity Reed-Muller Expression(PMPRM), which is a subclass of the Generalized Reed-Muller expression, is created, along with an efficient minimization algorithm. Since this new class contains many more forms than the the family of KRM, its minimum form is generally closer to the minimum ESOP, than is the minimum KRM form. It is a sub-class of the Generalized Reed-Muller Expansion, thus has better testability than other AND/EXOR circuits. The exact solution to the minimization of this new circuit form provides an upper-bound for the minimization of GRM expansion.

In this thesis, we prove that to calculate a PMPRM expansion from one of its adjacent polarity expansions, only one EXOR operation is needed. By calculating the adjacent polarity expansions one-by-one and searching all the PMPRM forms the minimum one can be found. A speedup approach allows us to find the exact minimum PMPRM without calculating all forms. The algorithm is explained by minimizing the 3-variable functions and is demonstrated by flow graphs.

With the introduction of a termwise complementary expansion diagram, a computerized algorithm for the calculation of any GRM expansion is presented. The exact

minimum GRM form can be obtained by an exhaustive search through all GRM forms. A heuristic minimization algorithm, which is designed to decrease the time complexity of the exact one, is also presented in this thesis. Instead of depending on the number of input variables, the computation time of this quasi-minimum algorithm depends mainly on the complexity of the input functions, thus can solve much larger problems.

VII.2 Future Work

The exact minimization algorithm for PMPRM and the quasi-minimum GRM minimization algorithm have been implemented in C programs and a set of benchmark functions have been tested. There are still several opportunities to further improve the speed, and especially, the space complexity of the programs. One method is to perform cube operations on disjoint cubes, instead of minimizing the coefficients, as is used in our algorithms. A possible extension would be also to develop the algorithms for multi-output functions. Further extension may be to deal with incompletely specified functions.

REFERENCE

[1] P.K. Liu and J.C. Muzio, "Boolean matrix transforms for the minimization of modulo-2 canonical expansions," *IEEE Trans. Comput.,* vol.C-41, pp.342-347, Mar.1992.

[2] A. Mukhopadhyay and G. Schmitz, "Minimization of exclusive OR and logic equivalence switching circuits," *IEEE Trans. Comput.*, vol.C-19, pp.132-140, Feb.1970.

[3] Robinson, J.P., and C.L. Yeh, "A Method for modulo-2 Minimization," *IEEE Trans.on Comput.,* vol.C-31, pp.800-801, Aug. 1982.

[4] M. Helliwell and M.A. Perkowski,"A fast algorithm to minimize multi-output mixed-polarity generalized Reed-Muller forms," in *Proc. of the 25th ACM/IEEE DAC*, pp.427-432, 1988.

[5] T.Sasao and P. Besslich, "On the complexity of mod-2 sum PLAs," *IEEE Trans. Comput.*, vol.C-39, pp.262-266, Feb. 1990.

[6] S.M. Reddy, "Easily testable realizations for logic functions," *IEEE Trans.Comput.,* vol.C-21, pp.1183-1188, Nov. 1972.

[7] K.K. Saluja and S.M. Reddy,"Fault detecting test sets for Reed-Muller canonic networks," *IEEE Trans. Comput.,*vol.C-24, pp.995-998, Oct.1975.

[8] R.Brayton, G. Hachtel, C. McMullen, and A. Sangiovanni-Vincentelli, *Logic Minimization Algorithms for VLSI Synthesis,* MA: Kluwer Academic Publisher, 1984.

[9] Ph.W. Besslich and M. W. Riege, "An efficient program for logic synthesis of mod-2 sum of expressions," in *Proc. of EURO-ASIC'91,* Paris, France, pp. 136-141, May 1991.

[10] T. Sasao, *Logic Synthesis and Optimization,* MA: Kluwer Academic Publishers, 1993.

[11] V. Kebschull and W. Rosenstiel, "Efficient graph-based computation and manipulation of functional decision diagrams," in *Proc. of EDAC*, pp. 278-282,1993.

[12] ATMEL Corpration, "Configurable Logic Design & Application Book," 1993.

[13] M.A. Perkowski and M. Chrzanowska-Jeske, "An exact algorithm to minimize mixed-radix Exclusive Sums of Products for incompletely specified Boolean functions," in *Proc. of IEEE ISCAS*, pp.1652-1655, 1990.

[14] D.H. Green, "Families of Reed-Muller canonical forms," *Int. J. Electron.* vol.70, pp.259-280, 1991.

[15] M. Davio, J.P. Deschamps, and A. Thayse, *Discrete and Switching Functions,* New York: McGraw-Hill, 1978.

[16] A. Sarabi and M.A. Perkowski, "Fast exact and quasi-minimal minimization of highly testable fixed-polarity AND/EXOR canonical networks," in *Proc. of 29th ACM/IEEE DAC*, pp.30-35, 1992.

[17] S. Kunclu, "Design of multioutput CMOS combinational logic circuits for robust testability," *IEEE Trans. on CAD*, vol.8, no.11, pp.1222-1226, 1989.

[18] Y.Z. Zhang and P.J.W. Rayer, "Minimization of Reed-Muller polynomials with fixed polarity," *IEE Proc.*, vol.131, part E, pp.177-186, Sept. 1984.

[19] M.A. Perkowski,"A fundamental theorem for EXOR circuits," in *Proc. of IFIP Workshop on Appl. of RM Expansion in Circuit Design,* Hamburg, pp.50-60 , 1993

[20] D.H. Green, "Reed-Muller canonical forms with mixed polarity and their manipulations," *IEE Proc.*, vol.137, part E, pp.103-113, Jan.1990.

[21] M. Cohn,"Inconsistent canonical forms of switching functions," *IRE Trans. Electron. Comput.*, vol.11, pp.284-285, 1962.

[22] M.A. Perkowski, L. Csanky, A. Sarabi and I.Schafer, "Fast minimization of mixed-polarity AND/EXOR canonical networks," in *Proc. of IEEE ICCD, Cambridge,* MA, pp.33-35, 1992.

[23] A. Sarabi and M.A. Perkowski, "Design for testability properties of AND/EXOR networks," in *Proc. of IFIP Workshop on Appl. of RM Expansion in Circuit Design,* Hamburg, pp.147-153, 1993.

[24] G. Papakonstantinou, "Minimization of modulo-2 sum of products," *IEEE Trans. on Comput.*, vol.C-28, pp.163-167, Feb. 1979.

[25] Ph.W. Besslich,"Efficient Computer Method for EXOR Logic Design," *Proc. IEE,* vol. 130, Part E, CDT, no. 6., pp.203-206,1983.

[26] X. Wu, X. Chen, and S.L. Hurst, "Mapping of Reed-Muller coefficients and the minimization of exclusive OR-switching functions," *IEE Proc. E, Comput. & Digital Tech.*, 1982, pp.15-20.

[27] H. Fleisher, M. Tavel, and J.Yeager, "A Computer Algorithm for Minimizing Reed-Muller Canonical Forms," *IEEE Trans. Comput.*, vol.C-36, No.2, pp.247-250, Feb. 1987.

[28] J.M. Saul,"An Improved Algorithm for the minimization of Mixed Polarity Reed-Muller Representations,"*Proc. Int. Conf. on Comput. Design: VLSI in Computers and Processors,* pp. 372-375, Sep.1990.

[29] D. Brand and T. Sasao, "On the minimization of AND-EXOR expressions,"*Int. Workshop on Logic Synthesis, Research Triangle Park,* NC, May 1991.

[30] S. Even, I. Kohavi and A. Paz,"On mimimal modulo-2 sum of products for switching functions," *IEEE Trans. on Comput.* vol. C-36, No. 2, Feb. 1967.

[31] D.H. Green, "Modern logic design", *Electronic Systems Engineering Series*, 1986.

[32] S.V. Kantopoulos, "Classification of exclusive-OR structures in the minimization of logic functions," *Int. J. Electron.*, vol.49, pp.115-129, 1980.

[33] T.Sasao, *Logic Synthesis and Optimization*, MA: Kluwer Academic Publisher, 1993

[34] M.A. Perkowski,"The generalized orthonormal expansion of functions with multiple-valued inputs and some of its applications," *Proc. of 22nd International Symposium on Multiple-Valued Logic,* May 1992, pp. 442-450.

[35] T. Sasao, "Easily testable realization for generalized Reed-Muller expressions."

[36] N. Song, "Minimization of exclusive sum of products expressions for multiple-valued input incompletely specified functions," *Master thesis,* Portland State University, 1994.